Biomechanical Modeling of the Human Body for Application to Wheelchair Seating Systems

Undergraduate Honors Thesis

Presented in Partial Fulfillment of the Requirements for
Graduation with Honors Research Distinction in the
Department of Mechanical Engineering at
The Ohio State University

By:
Ryan William Letcher

Advisors:
Sandra A. Metzler, D.Sc., P.E.
Carmen P. DiGiovine, Ph.D.

April 2016

ABSTRACT

People who have physical and/or mental disabilities often require wheelchairs to conduct everyday activities. Sitting in a wheelchair for an extended period of time can cause physical injuries such as pressure ulcers, spinal degeneration, and other biomechanical health issues. Assistive seating systems are integrated into wheelchair designs with the intent that the user has correct seated posture, which generally prevents these injuries. However, it is difficult to accurately predict how a seating system is going to affect an individual user due to the multiple conditions that may affect the user. The purpose of this research is to conduct a preliminary biomechanical analysis of the human body when in a seated position, and to specifically apply this analysis to individuals with disabilities. The initial research focuses on the development of static equilibrium equations and models to calculate forces and pressures on the body, the implementation of those models in MATLAB software such that they can be easily integrated into existing biomechanical simulation software, and the development of a user interface for use by clinicians to report relevant data. Thus far, MATLAB code was written to model the human body as rigid body segments using data averaged over multiple studies of individuals. The code receives inputs of the overall height and weight of a potential wheelchair user and body angles to represent the orientation of the user in the wheelchair. The code outputs a pictorial representation of the user and the point forces associated with postural supports that act on him or her. Future work includes the elimination of included assumptions in the model and implementation of the model into OpenSim software. This research will reduce the amount of time clinicians spend to properly fit a wheelchair to its user and will save wheelchair users money and prevent future injuries by helping them choose the correct seating system the first time.

ACKNOWLEDGEMENTS

This research was supported by the College of Engineering Undergraduate Research Scholarship Fund. I would like to thank my advisors Dr. Sandra Metzler and Dr. Carmen DiGiovine for providing the opportunity for me to conduct this research and for all of the help they offered.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

People with disabilities often require the use of a wheelchair in order to move around on a daily basis. Because of their need to use a wheelchair for mobility, they often find themselves seated in their wheelchairs for hours at a time without the opportunity to sit on a different surface or even to reposition their body. This lack of postural change forces wheelchair users to desire a more comfortable seated position, regardless of the strain that this position induces on certain segments of the body. It is possible that by maintaining this body posture for too long, a wheelchair user is likely to develop postural deformities including, but not limited to, scoliosis, kyphosis, and lordosis, or to develop pressure ulcers, which are defined as "areas of skin that break down when something keeps rubbing or pressing against the skin" [1]. These issues arise due to misalignment of the spine within the wheelchair, tilting of the pelvis in relation to the spine and legs, or having a poorly positioned center of gravity.

The center of gravity (COG) of an object is "the location of the center of mass in the vertical direction," where the center of mass (COM) is defined as "the net location of the center of mass in three-dimensional space" [2]. Essentially, the COM represents a point location where the force of gravity can be modeled. For any object resting on a base of support, the object will not tip over so long as the COG of the object is maintained within the boundaries of the base of support. Humans subconsciously practice this principle every day when standing upright and maintaining balance. It is possible, however, for the COM of an object to be outside of the boundaries of the object itself. For example, as seen in Figure 1, due to the nature of the flexibility of the human body, the COM of a person who is bent over with their hands on the

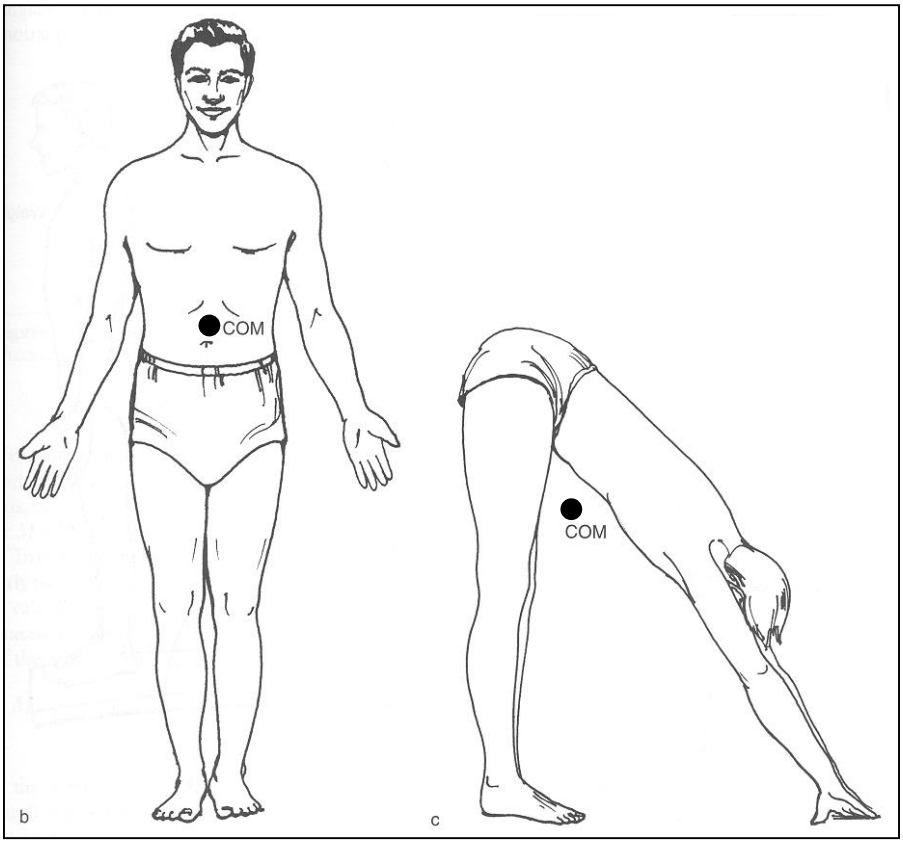

Figure 1: Location of COM of human body in different positions *[7]*.

ground will be located outside of the body. It should be noted that in the picture on the right of Figure 1, the COG of the body is within the base of support, which is the area between the heel of the feet and tips of the fingers, and thus, the body will not tip over when in this position. The principle of the COG of a body is important in wheelchair use because the COG must be maintained within the bounds of the wheelchair to prevent the user from falling out of the wheelchair.

In order to maintain proper seated posture in a wheelchair and to minimize user effort, a clinician will attached assistive postural supports to the wheelchair. These postural supports range from common wheelchair attachments such as sitting cushions, backrests and footrests to more advanced attachments such as lateral trunk supports, lateral thigh supports, seat belts, etc. Each support has a unique placement on the wheelchair that can be adjusted by a clinician in

order to enhance the comfort of the user and to ensure proper posture of the patient. These supports vary in size, softness, material, and curvature, and can be custom made to fit a specific patient's needs. By implementing these supports onto wheelchairs, the supports induce a force on the user's body that over time can create irritation on the body if the force is too high.

1.1 The Wheelchair Fitting Process

When a patient needs to be fitted to a wheelchair for the first time, a clinician will take length measurements of certain body parts. These measurements are utilized to determine the properly sized wheelchair for the patient to fit comfortably in. Additionally, the clinician will conduct a mat examination, which is where the patient lies supine on a flat surface and the clinician moves different body parts of the patient in order to test the patient's range of motion for various joint and body segments, such as hip flexion and knee extension. Knowing these ranges of motion is important for the clinician because it allows the clinician to know what positions the patient will or will not be able to sit in while in the wheelchair. Based upon this information, the clinician will suggest and order various postural supports that he or she believes will give the patient the best seated posture to ensure comfortability while reducing the likelihood of postural deformity development.

After viewing wheelchair fittings and speaking with clinicians, it was determined that there are many issues with this process:

1) The methods for measuring the length of body parts are inaccurate and can cause incorrectly sized postural supports or wheelchairs to be ordered for the patient, which makes the process for obtaining the perfect wheelchair fit for the patient more costly and time consuming.

2) The determination of the ranges of motion of the patient's joints is mostly qualitative, which means each clinician has his or her own scale with which to determine the flexibility of a patient.
3) Clinicians must use their prior knowledge of different wheelchair designs and different postural supports in order to "guess" which total seating system will work best for a patient.
4) Clinicians are generally unsure of how much force each support will exert on a patient, and how having multiple supports pushing against a patient will affect the patient's posture.

Once a patient's wheelchair and postural supports have been ordered, received, and assembled by the clinician, the patient will come back in to test out the wheelchair and make adjustments as needed. This process becomes iterative if the adjustments made during a visit do not turn out to be well suited for the patient long term. The process can be both time consuming and costly for the patient, which in many situations, the patient cannot afford to have happen.

One tool used by clinicians in order to verify comfortability and proper posture is a pressure mat. A pressure mat, which is shown in Figure 2, is simply a thin mat with internal sensors that "measure the pressure distribution of a human body on support surfaces such as seats, mattresses, cushions, and backrests… with minimal interference of the support surface" [3]. In this case, the mat is used on the underside of a patient sitting in a wheelchair. The pressure mat connects to a computer via USB and outputs the pressure of the cushion on the patient's

Figure 2: Pressure mat *[3]*.

body using a software program specific to the mat. This output can be seen in Figure 3. The clinician uses this view of the pressure to identify and reduce pressure spikes (or small areas of high pressure) because these spikes are uncomfortable for most users and generally place a force high enough on the patient's body to create a pressure ulcer. Eliminating or reducing these pressure spikes involves attaching additional postural supports to the wheelchair, moving the positions of the current postural supports, changing to softer or harder postural supports, or a combination of the three.

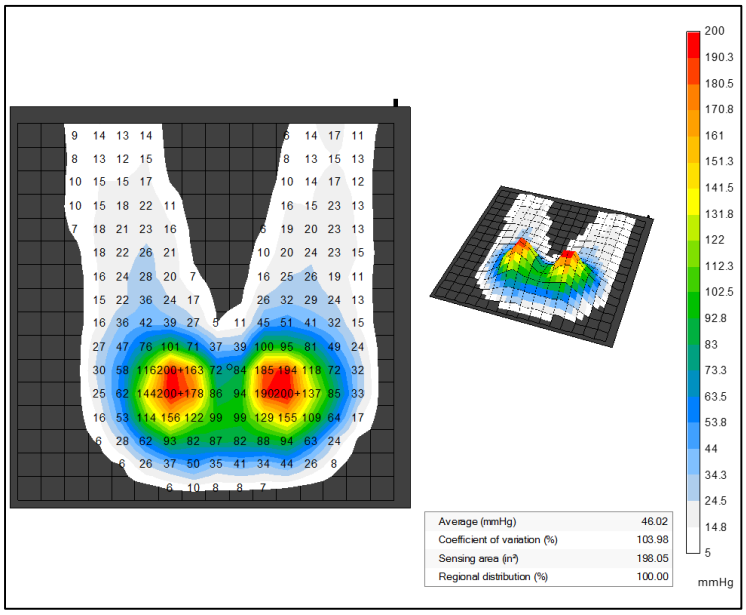

Figure 3: User interface of a pressure mat.

1.2 Purpose of Research

The purpose of this research is: 1) to conduct a biomechanical analysis on a human body in a seated position, 2) to analyze the effects of postural supports on the body's alignment in a wheelchair, and 3) to develop a preliminary biomechanical model that can be utilized in the development of a user friendly tool for clinicians that will facilitate and improve the wheelchair fitting process as it is applied to people with disabilities.

1.3 Significance of Research

In 2002, the US census estimated that over 2.8 million people in the United States of America used a wheelchair on a daily basis [4]. Due to the large number of wheelchair users, it is difficult for clinicians to spend a lot of time with one patient correcting the problems found on the wheelchair. This research will reduce the amount of time that clinicians must spend to properly fit a new wheelchair to its user by generating a biomechanical analysis of the human body and determining the forces associated with postural supports, which are the first steps in the development of a quick and easy-to-use clinical software program that will more accurately predict the effect a wheelchair has on its user's body and posture. This reduction in time will ultimately save patients time and money by choosing the correct wheelchair the first time and by preventing future injuries to patients due to bad posture. Additionally, this clinical tool will enhance the quality of life for a wheelchair user by ensuring the locations of the postural supports allow for more range of motion and comfortability for the user.

1.4 Outline of Thesis

This thesis describes the work completed to model the human body in a seated position and understand the effect of postural supports on the amount of force applied to the body. The first chapter of the thesis provides background information about the current wheelchair fitting

process and reasons why this research is important. The second chapter describes the methods used in creating the model. The third chapter provides results of the current output of the model and discusses the relevancy of the project. The fourth and final chapter provides direction for future work on this topic. Appendices can be found at the end of the document.

CHAPTER 2: METHODS

The biomechanical analysis of the human body was conducted by developing free body diagrams of the body in various seated positions. This allowed forces from the supports to be applied to the body and calculated by using the following equations:

$$\sum F_x = 0$$

$$\sum F_y = 0$$

$$\sum T_A = 0$$

where F_x represents a force in the x-direction, F_y represents a force in the y-direction, and T_A represents the torque about an arbitrary point A within the model.

2.1 Body Segment Model

In order to model the entire body, the body had to be broken down into rigid body segments and drawn together as straight lines. The body segments used were: the head, the trunk, two upper arms, two forearms, two hands, two thighs, two shanks, and two feet. Each body segment had its own approximate length, location on the body, weight, and COG location. The length of each body segment and its respective location on the body was calculated as a percentage of the total height of the human body inputted into the model. These percentages were determined by dividing the body segment lengths and locations of a 50^{th} percentile male by

the overall height of a 50th percentile male. The data for the body segment lengths and locations of a 50th percentile male were pulled from the widely excepted book titled The Measure of Man and Woman: Human Factors in Design [5]. A diagram of a 50th percentile male from this book can be seen in Figures 4a and 4b. The weight of each body segment was calculated as a

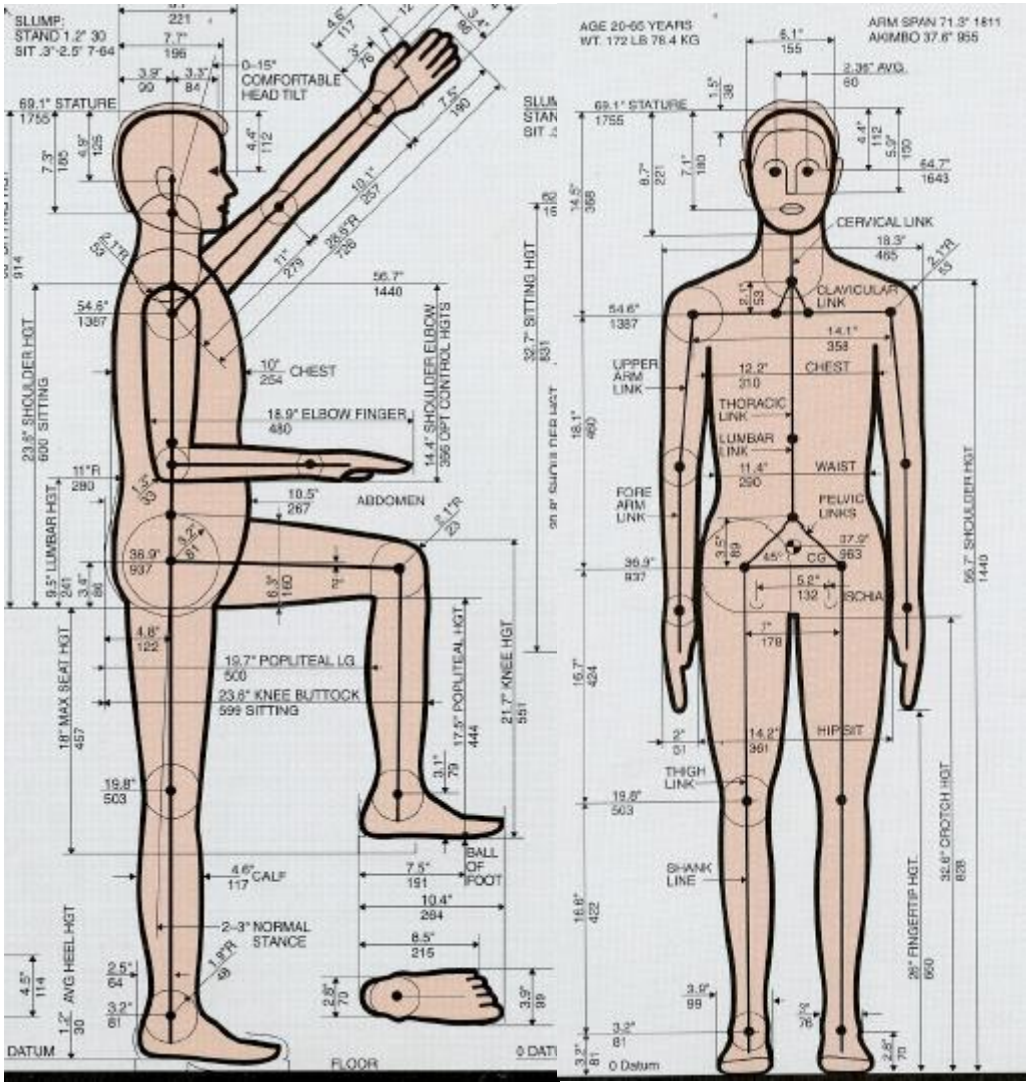

Figure 4: (a) Side view of 50th percentile male, (b) Front view of 50th percentile male *[5]*.

percentage of the total weight of the body, while the COG location was calculated as a percentage of the length of each body segment from the proximal end. Both of these sets of percentages were pulled from data found in the book Biomechanics and Motor Control of Human Movement [2] and can be found in Figure 5.

Segment	Definition	Segment Weight/Total Body Weight	Center of Mass/ Segment Length Proximal	Center of Mass/ Segment Length Distal	Radius of Gyration/ Segment Length C of G Proximal	Radius of Gyration/ Segment Length Distal	Density
Hand	Wrist axis/knuckle II middle finger	0.006 M	0.506	0.494 P	0.297	0.587	0.577 M	1.16
Forearm	Elbow axis/ulnar styloid	0.016 M	0.430	0.570 P	0.303	0.526	0.647 M	1.13
Upper arm	Glenohumeral axis/elbow axis	0.028 M	0.436	0.564 P	0.322	0.542	0.645 M	1.07
Forearm and hand	Elbow axis/ulnar styloid	0.022 M	0.682	0.318 P	0.468	0.827	0.565 P	1.14
Total arm	Glenohumeral joint/ulnar styloid	0.050 M	0.530	0.470 P	0.368	0.645	0.596 P	1.11
Foot	Lateral malleolus/head metatarsal II	0.0145 M	0.50	0.50 P	0.475	0.690	0.690 P	1.10
Leg	Femoral condyles/medial malleolus	0.0465 M	0.433	0.567 P	0.302	0.528	0.643 M	1.09
Thigh	Greater trochanter/femoral condyles	0.100 M	0.433	0.567 P	0.323	0.540	0.653 M	1.05
Foot and leg	Femoral condyles/medial malleolus	0.061 M	0.606	0.394 P	0.416	0.735	0.572 P	1.09
Total leg	Greater trochanter/medial malleolus	0.161 M	0.447	0.553 P	0.326	0.560	0.650 P	1.06
Head and neck	C7–T1 and 1st rib/ear canal	0.081 M	1.000	— PC	0.495	0.116	— PC	1.11
Shoulder mass	Sternoclavicular joint/glenohumeral axis	—	0.712	0.288	—	—	—	1.04
Thorax	C7–T1/T12–L1 and diaphragm*	0.216 PC	0.82	0.18	—	—	—	0.92
Abdomen	T12–L1/L4–L5*	0.139 LC	0.44	0.56	—	—	—	—
Pelvis	L4–L5/greater trochanter*	0.142 LC	0.105	0.895	—	—	—	—
Thorax and abdomen	C7–T1/L4–L5*	0.355 LC	0.63	0.37	—	—	—	—
Abdomen and pelvis	T12–L1/greater trochanter*	0.281 PC	0.27	0.73	—	—	—	1.01
Trunk	Greater trochanter/glenohumeral joint*	0.497 M	0.50	0.50	—	—	—	1.03
Trunk head neck	Greater trochanter/glenohumeral joint*	0.578 MC	0.66	0.34 P	0.503	0.830	0.607 M	—
Head, arms, and trunk (HAT)	Greater trochanter/glenohumeral joint*	0.678 MC	0.626	0.374 PC	0.496	0.798	0.621 PC	—
HAT	Greater trochanter/mid rib	0.678	1.142	—	0.903	1.456	—	—

Figure 5: Anthropometric data of relative masses and COG locations of body segments *[2]*.

2.2 Modeling Software (MATLAB)

A few software programs were investigated to conduct this biomechanical analysis. An easy-to-use biomechanical, 3D software program called OpenSim was first investigated due to its extensive use at The Ohio State University by both faculty and students in the biomedical field. Issues arose with this software program, however, because OpenSim is better used for modeling dynamic simulations of the human body in order to model and predict joint torques and muscle activations within the body, as opposed to determining static, external forces on the body due to other objects pressing against the body.

Another software program that was investigated was MATLAB, also due to its extensive use at The Ohio State University by engineering students and faculty. MATLAB is an easy-to-code programming language that has the capability of performing many operations quickly and effectively. MathWorks, the creator of MATLAB, says on their website that “the MATLAB platform is optimized for solving engineering and scientific problems. The matrix-based

MATLAB language is the world's most natural way to express computational mathematics. Built-in graphics make it easy to visualize and gain insights from data" [6]. It was decided to use MATLAB for modeling the human body because it offered the most straight forward interface and contained all of the tools necessary to calculate external forces.

2.3 Description of MATLAB Code

The final MATLAB codes written for this research, which can be found in Appendix A, were split into two views of the body: the side view which allowed the cushion, the backrest and the footrests to be modeled and the front view which allowed the cushion, the footrests and the lateral trunk supports to be modeled. Both codes were run with inputs of the overall height and weight of the person and specific joint angles that the program user desired to see the body positioned in. These joint angles included the angle at the pelvis, at the knee, at the ankle, at the shoulders, at the elbows, and at the wrists. The angles at the pelvis and at the shoulders included inputs for both the sagittal plane angles (side view) and the frontal plane angles (front view). A diagram of the joint angles can be seen in Figures 6a and 6b.

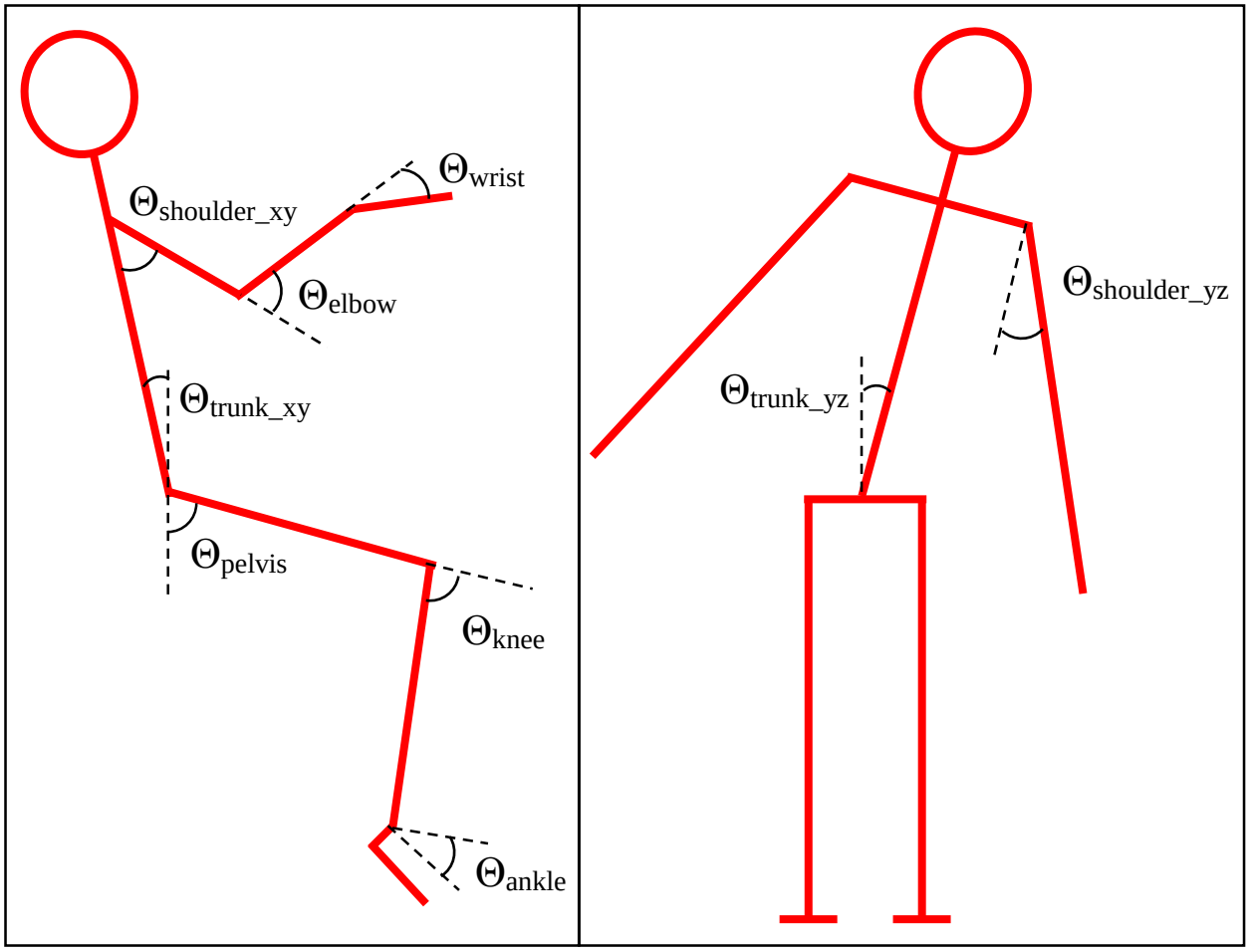

Figure 6: (a) Side view input angles, (b) Front view input angles.

After reading in these values, the codes calculated the lengths, weights, and locations of the body segments based upon the percentage data mentioned in Section 2.1 of this thesis. For example, the calculation for the length and weight of the upperarm can be seen in the following equations:

$$\boldsymbol{upperarmlength} = 0.1592 * height$$

$$\boldsymbol{upperarmweight} = 0.028 * weight$$

For a male who is 69.1 inches tall and weighs 172 pounds, which represents a 50th percentile male, the length of the upper arm would be 11 inches and the weight of the upper arm would be 4.82 pounds. The codes then calculated the point location of the COG of each body segment in

the respective plane using the percentage data for the COG of individual body segments. The equation used for this calculation based all points around a stationary pelvis, which is located at:

$$x = 0$$

$$y = 0.534 * height = \boldsymbol{pelvislocation}$$

$$z = 0$$

Thus, in the xy-plane (side view), the equations for the x-coordinate and the y-coordinate of the COG of the upper arm are:

$$\boldsymbol{upperarm_{COGx}} = 0.436 * upperarmlength * \sin\left(\theta_{shoulder_{xy}} + \theta_{trunk_{xy}}\right) - trunklength * \sin\left(\theta_{trunk_{xy}}\right)$$

$$\boldsymbol{upperarm_{COGy}} = pelvislocation + trunklength * \cos\left(\theta_{trunk_{xy}}\right) - 0.436 * upperarmlength * \cos(\theta_{shoulder_{xy}} + \theta_{trunk_{xy}})$$

Next, using the COG locations of all body segments and the following equation, the codes calculated the COG of the entire body:

$$\boldsymbol{wholebody_{COGx}} = \frac{1}{weight} * \sum segment\ mass * segment\ COG$$

$$= \frac{1}{weight} * \sum(headweight * head_{COGx} + trunkweight * trunk_{COGx} + upperarmweight * upperarm_{COGx} + forearmweight * forearm_{COGx} + \cdots)$$

$$\boldsymbol{wholebody_{COGy}} = \frac{1}{weight} * \sum segment\ mass * segment\ COG$$

$$= \frac{1}{weight} * \sum(headweight * head_{COGy} + trunkweight * trunk_{COGy} + upperarmweight * upperarm_{COGy} + forearmweight * forearm_{COGy} + \cdots)$$

This equation was also used to calculate the COG of the upper portion of the body, which included the masses of the trunk, the head, and all parts of both arms. After determining these COG locations, the codes plotted the body segments onto a gridded figure window and identified all of the calculated COG locations using small, colored circles. This was completed by identifying the proximal and distal end points of each body segment and plotting a straight line between them. For example, to draw the upper arm, the location of the shoulder and the elbow had to be identified, which was done using the following equations:

$$\boldsymbol{upperarm_{shoulder_x}} = -trunklength * \sin\left(\theta_{trunk_{xy}}\right)$$

$$\boldsymbol{upperarm_{shoulder_y}} = pelvislocation + trunklength * \cos\left(\theta_{trunk_{xy}}\right)$$

$$\boldsymbol{upperarm_{elbow_x}} = -trunklength * \sin\left(\theta_{trunk_{xy}}\right) + upperarmlength * \sin(\theta_{trunk_{xy}} + \theta_{shoulder_{xy}})$$

$$\boldsymbol{upperarm_{elbow_y}} = pelvislocation + trunklength * \cos\left(\theta_{trunk_{xy}}\right) - upperarmlength * \cos(\theta_{trunk_{xy}} + \theta_{shoulder_{xy}})$$

Then, the side view code calculated the force of the backrest, the cushion, and the footrests. For each of these calculations, the code first checked to see which support each body segment was above and created a total weight to be applied to the body segment. For example, if the head, trunk, and upperarm are supported by the backrest, then:

$$\boldsymbol{bodyweight_{backrest}} = headweight + trunkweight + upperarmweight$$

$$\boldsymbol{backrest\ normal\ force} = bodyweight_{backrest} * \sin(\theta_{trunk_{xy}})$$

The front view code performed the same procedures as the side view code, however, it calculated the force of the cushion, the footrests, and the lateral trunk supports instead. Finally, the codes calculated the point locations at which each of the respective supports acted on the body in the

inputted orientation and plotted those point locations as circles on the gridded figure window. The locations were determined by calculating the total torque due to each body segment's weight and then using the sum of torques equation below:

$$\boldsymbol{torque_{backrest}} = headweight * head_{COGx} + trunkweight * trunk_{COGx} + upperarmweight * upperarm\_COGx$$

$$\boldsymbol{backrest\ force\ length} = -\frac{torque_{backrest}}{backrest\ normal\ force}$$

$$\boldsymbol{backrest\ location_x} = -backrest\ force\ length * \sin(\theta_{trunk_{xy}})$$

$$\boldsymbol{backrest\ location_y} = pelvislocation + backrest\ force\ length * cos(\theta_{trunk_{xy}})$$

2.4 Weight Test

If a body segment was in direct contact with a wheelchair support, then the entire weight of that body segment acted on the support. However, if a body segment was not touching a support, such as the hand when the arms are fully extended forward, it was unclear as to where the weight of that segment would act in relation to the entire wheelchair. In order to understand the effect of these body segments, a weight test was conducted with the body in various positions. This test included having the participant sit upright on top of a scale with his arms fully extended forward, with his feet propped up on a stool, and without having his back pressed up against any object. This particular setup for the test can be seen in Figure 7. The readout from the scale was recorded. The scale was then placed underneath the feet and on top of the stool and the test was run again with the same setup of the arms fully extended. The readout from the scale was once again recorded. This process was repeated with the participant's arms hanging freely at his sides and not touching any objects. The readouts from the scale were recorded for the scale being underneath the participant's underside and underneath the participant's feet.

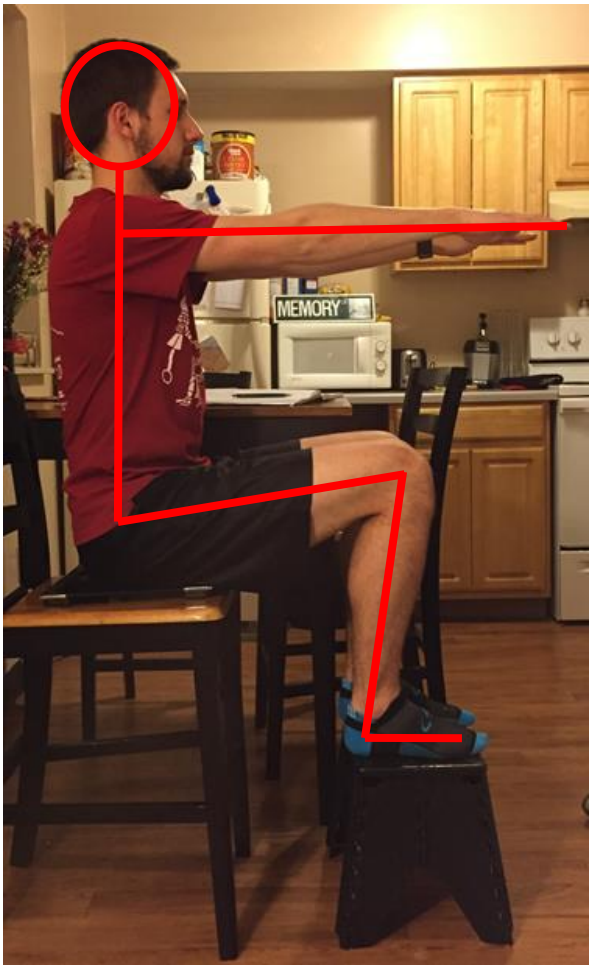

Figure 7: Test setup with arms fully extended and scale underneath participant.

CHAPTER 3: RESULTS AND DISCUSSION

3.1 Weight Test

When the participant had his arms fully extended, the scale underneath the underside of the participant read 124.8 pounds while the scale underneath the feet of the participant read 45.4 pounds. When the participant had his arms hanging by his sides, the scale underneath the underside of the participant read 135.4 pounds while the scale underneath the feet of the participant read 28.2 pounds. When the participant's arms were fully extended, the COGs of the hands were located over the scale under the feet. These results allowed the assumption to be

made that the location of the COG of a body segment that is not in direct contact with a support determines which support the weight of that particular body support will act upon. For example, if the COGs of the hands of the modeled body are located over the footrest, then the calculation for the force from the footrest will include the weight of the hands.

3.2 MATLAB Program Output

The output of the MATLAB codes provides the program's user with the numerical values for the point forces associated with each support in each view by outputting the values to the command window of MATLAB for easy viewing. Additionally, a pictorial representation of the free body diagram of the human body in the inputted position is generated on a plot with gridlines to represent the special coordinates of each body segment. The following legend, labeled as Figure 8, identifies what each symbol in the outputted plots represents:

Red Dot = COG of individual body segments
Purple Dot = COG of entire body
Gray Dot = center of pressure of backrest/lateral trunk supports
Green Dot = center of pressure of cushion
Brown Dot = center of pressure of footrest

Figure 8: Legend for MATLAB free body diagrams.

The models shown in the following subsections all represent a 50^{th} percentile male who has an overall height of 69.1 inches and an overall weight of 172 pounds, as determined by The Measure of Man and Woman [5].

3.2.1 Side View: Seated Upright

A normal position for a person to be seated in is directly upright if he or she is seated on a flat surface with no backrest and with the feet planted on a surface parallel to the sitting surface. The following figure shows the side view of the body segment representation seated in this position:

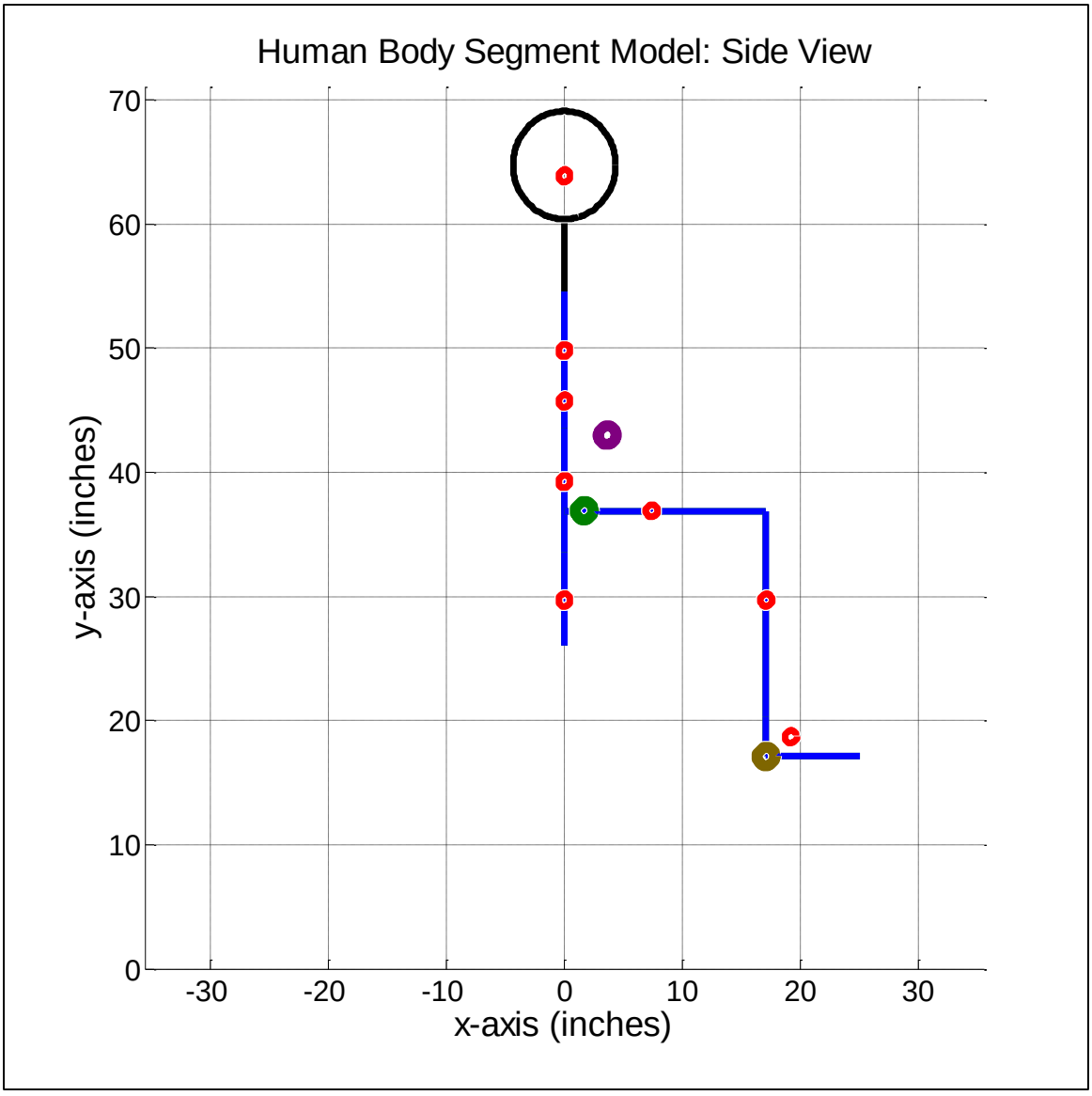

Figure 9: Side view representation of a seated upright human body.

Since there is no backrest in this orientation, the output for the backrest force reads 0 pounds. Due to a lack of a backrest force, the cushion friction force, which acts parallel to the plane of

the cushion and in this case is horizontal, also reads 0 pounds. The cushion normal force, which acts perpendicular to the plane of the cushion and in this case is vertical, reads 151.0 pounds. This value makes sense because it is the total weight of the upper body (head, trunk, and arms) plus the weight of the thighs, all of which are located above the cushion and are solely supported by the cushion. The footrest force reads 21.0 pounds, which also makes sense since it supports the weight of the shanks and the feet.

3.2.2 Front View: Seated Upright

The following figure shows the same bodily orientation as Section 3.2.1 but displays the front view of the body segment model:

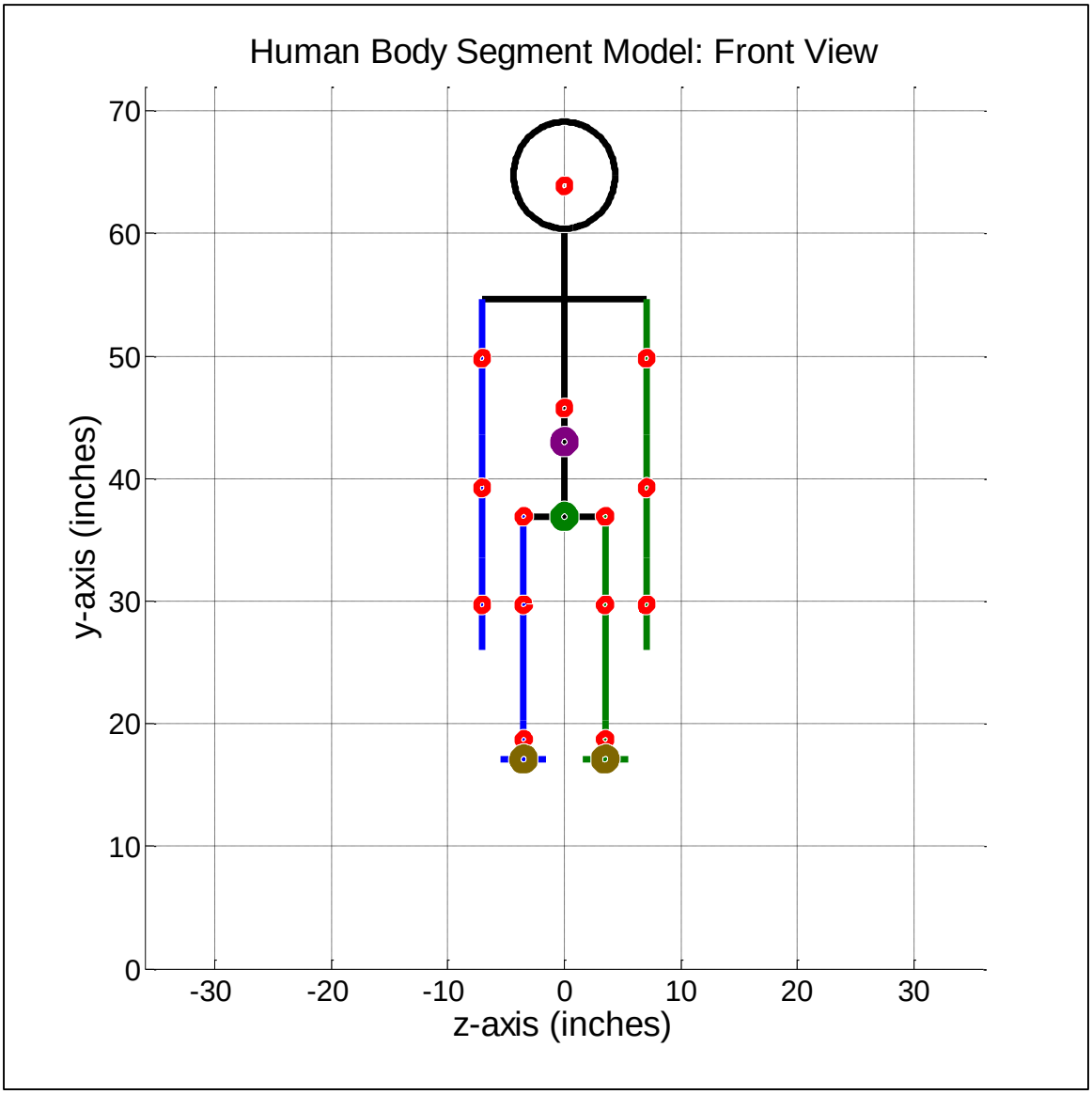

Figure 10: Front view representation of a seated upright human body.

Since there are no lateral trunk supports required for a person to maintain this position, the outputs for the forces of both the left and right trunk supports read 0 pounds. Due to a lack of trunk support forces, the cushion normal force in this view also reads 0 pounds. The cushion normal force, which acts perpendicular to the plane of the cushion and in this case is vertical,

reads 151.0 pounds. The left footrest reads 10.5 pounds and the right footrest reads 10.5 pounds. The two forces are equal because each footrest supports the same set of body parts: the shank and the foot. The values for the cushion and footrests make sense because the model has the same bodily orientation as the previous section and thus should read the same forces that were calculated in the side view.

3.2.3 Side View: 10° Recline

When sitting for extended periods of time, people in wheelchairs, and most people in general, prefer to recline their chairs such that the backrest holds some of their weight and the force on their underside due to the cushion is reduced. The following figure shows the side view representation of this reclined position with the arms maintaining a collinear position with the trunk:

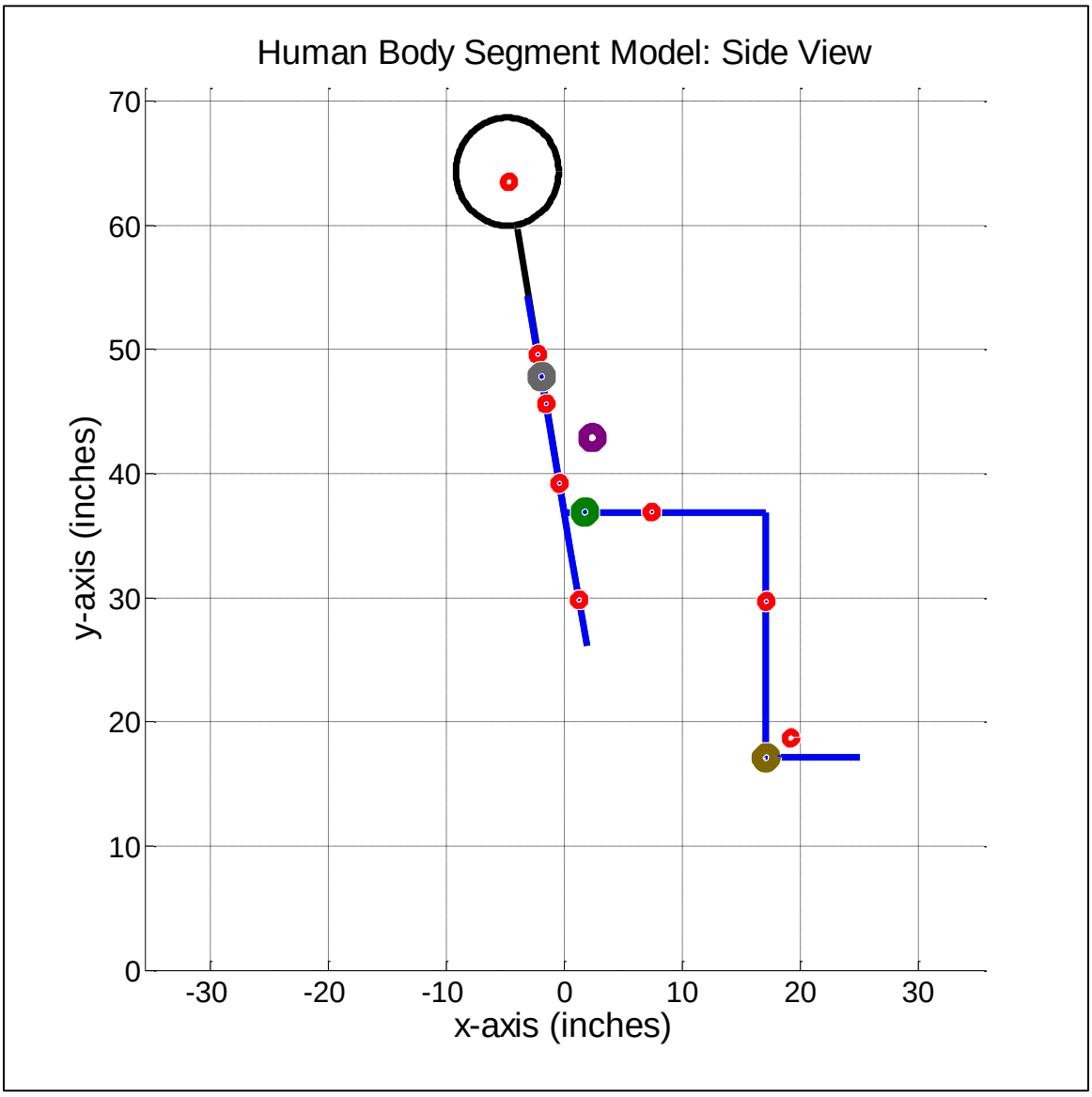

Figure 11: Side view representation of a 10° reclined human body.

In this orientation, a backrest force is calculated and read to be 19.9 pounds. As mentioned earlier, this backrest force helps reduce the cushion normal force from 151.0 pounds from the seated upright position down to 147.6 pounds. However, this backrest force pushes the wheelchair user forward so a cushion friction force must be present to prevent the user from moving forward and sliding out of the chair. In this case, the cushion friction force reads 19.6 pounds. This friction force, if too high, is bad for a wheelchair user because the user can develop pressure ulcers on the underside of their body.

3.2.4 Side View: 10° Tilt

In order to reduce this friction force, a clinician will suggest that a wheelchair user tilt the entire wheelchair back as opposed to only reclining the backrest of the chair. The following figure displays the body segment representation of this tilt from the side view:

Figure 12: Side view representation of a 10° tilted human body.

Per the output of the MATLAB program, the cushion friction force is reduced from 19.6 pounds to 6.3 pounds, which not only helps in the prevention of pressure ulcers but also increases the comfortability of the wheelchair user. The backrest force in the tilting orientation remains the

same as that in the reclining orientation, which is 19.9 pounds. The cushion normal force increases from 147.6 pounds to 148.7 pounds because the plane of the cushion is now at an angle and helps to offset the force produced by the backrest. Additionally, with the shanks and feet at an angle, a footrest behind the legs must be used to maintain the position of the lower legs. The force from the footrest behind the legs reads 3.6 pounds and the force from the footrest under the feet reads 20.7 pounds.

3.2.5 Front View: 10° Lean

Another common posture for wheelchair users and people sitting in common chairs is to lean sideways and support their body weight with their arm on an armrest. This is bad for a wheelchair user because it puts strain on the shoulder joint and causes the spine to be misaligned with the pelvis. To prevent this behavior, a clinician will place lateral trunk supports along the sides of a wheelchair user's chest in order to maintain the upright posture of the trunk. The following figure represents a body leaning against a lateral trunk support in the front view:

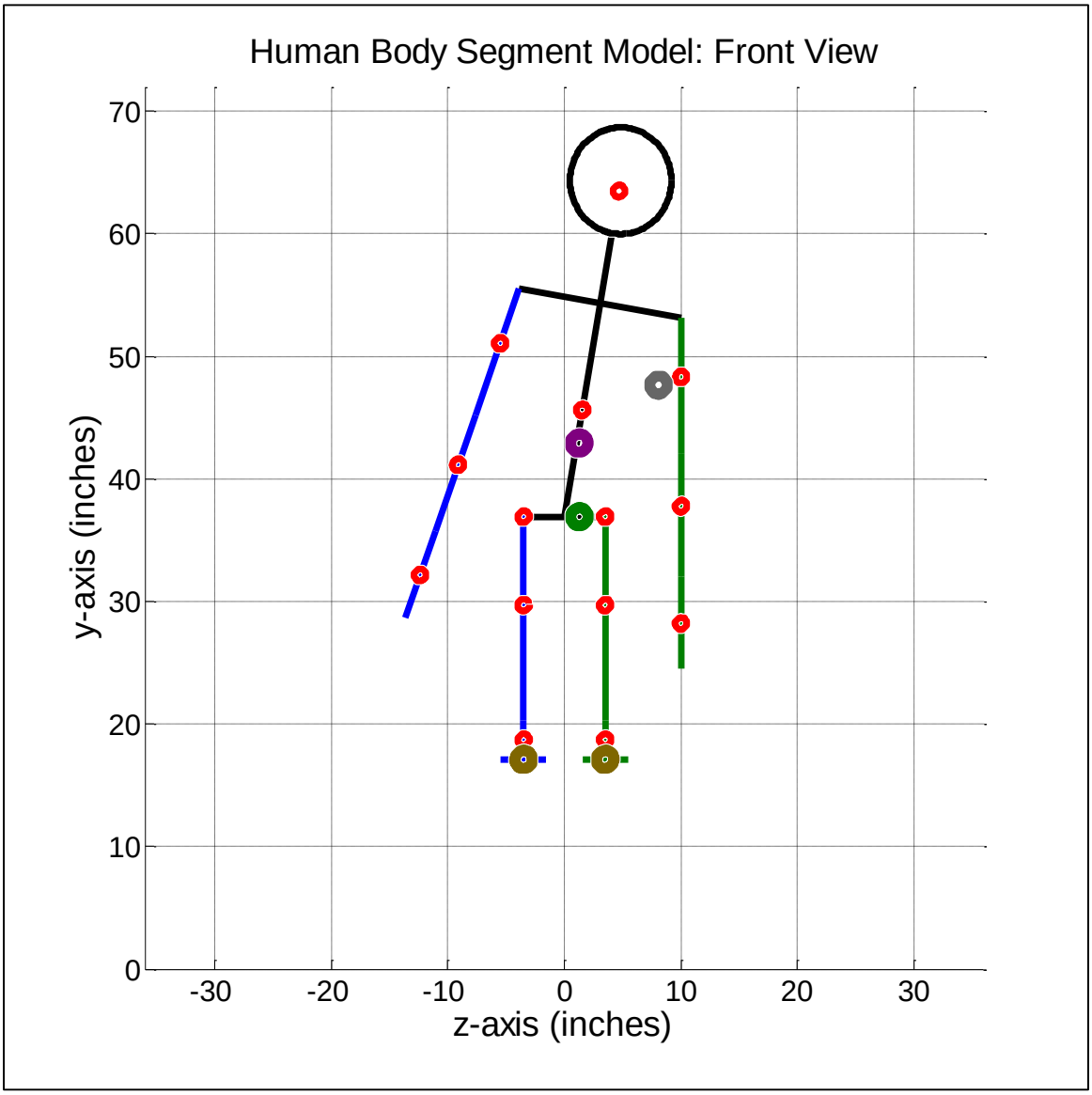

Figure 13: Front view representation of a 10° leaned human body.

Since the model is leaning to its left side, the left lateral trunk support reads 20.3 pounds while the right lateral trunk support reads 0 pounds because none of the body segments are touching it. The cushion normal force is slightly reduced from 151.0 pounds to 147.5 pounds when compared to the sitting upright orientation. However, just like with the introduction of the backrest force, the lateral trunk support force requires a cushion friction force to balance it. This cushion friction force reads 19.9 pounds. In order to reduce that friction force, a clinician will typically attach either a lateral thigh support or a seatbelt to hold the pelvis in place more securely.

3.3 Support Forces

The forces outputted by the MATLAB program represent the point forces associated with each of the supports. The point force acts as the center of the pressure distribution across the entire support; but in reality, the support does not act solely on one point of the body, but rather the force is distributed across the contact area between the support and the body. For now, the pressure distribution is modeled as a point force to simplify the governing equation of the sum of the torques being equal to zero. The average pressure associated with each support can be calculated by dividing the point force by the contact area; however, this proves to be difficult because the contact area differs for every person and is extremely difficult for a clinician to measure.

3.4 Model Assumptions

Many assumptions were made in the development of the model due to limitations in current research on the human body and due to the uniqueness of every human being. Since all of the data were taken from a 50^{th} percentile male, this model currently cannot be applied to those near the extremes of height or weight of the population, e.g. 5^{th} or 95^{th} percentile, or to those whose body shape or bodily distribution of weight deviate highly from that of a 50^{th} percentile male. Additionally, the model cannot be applied to those with postural deformities because the body is modeled as rigid straight lines, as opposed to, for example, the curved nature of a healthy or deformed spine. Some overarching assumptions made in the model are:

1) The footrest has no friction force, but rather acts as two different footrests with only normal forces, where one footrest is behind the shank and the other footrest is underneath the feet;

2) The feet are always at 90° with respect to the shank, which allows the two modeled footrests to always be perpendicular to one another;
3) The force from the backrest has no friction component and thus the cushion must have a friction force to prevent the wheelchair user from sliding off of the front of the wheelchair;
4) The forces from the lateral trunk supports have no friction component and thus the cushion must have a friction force to prevent the wheelchair user from sliding off of the side of the wheelchair.

All of these assumptions had to be made in order to keep the free body diagram of the body segments statically determinate, which means the free body diagram has the same number of equations as unknown variables such that each variable, in this case forces and locations, can be solved for.

CHAPTER 4: FUTURE WORK

This project is still in the beginning stages and as work was being completed on it, new directions for future research in the same area began to arise. Currently, the side view and the front view of the model are two different MATLAB codes that are run independently of each other. Within each code, certain assumptions had to be made regarding the orientation of the body in the other plane. It is possible to combine both models and plane angles into a single code such that any orientation, regardless of the impossibility of the posture, can be correctly modeled. This will create a better understanding of the relationships between all of the supports that would be placed on a wheelchair at one time and give a clinician the opportunity to view how the patient would/should look in each plane.

Another area of improvement in the model includes the elimination of the assumptions listed in Section 3.4. Most of the assumptions described the lack of a friction force for certain supports because the model would become statically indeterminate. Instead, a constant static coefficient of friction between the person in the wheelchair and the respective support could be implemented into the model to create a relationship between the friction force and the normal force of the supports. Using this relationship should allow the model to remain statically determinate while more accurately predicting support forces.

Finally, a future project could involve modeling the human body in a 3D biomechanical software program called OpenSim. Utilizing OpenSim will allow for internal joint torques and muscle activations to be determined such that this work can be extended to people with disabilities. OpenSim could also show the program user how the position of the body changes as the muscles begin to relax over time.

REFERENCES

[1] "U.S. National Library of Medicine," 20 November 2012. [Online]. Available: https://www.nlm.nih.gov/medlineplus/ency/article/007071.htm.

[2] D. A. Winter, Biomechanics and Motor Control of Human Movement, Hoboken: John Wiley & Sons, Inc., 2009.

[3] "Tekscan," [Online]. Available: https://www.tekscan.com/products-solutions/systems/matscan.

[4] "Disabled World," 26 February 2015. [Online]. Available: http://www.disabled-world.com/assistivedevices/mobility/wheelchairs/.

[5] A. R. Tilley, The Measure of Man and Woman: Human Factors in Design, New York: Whitney Library of Design, 1993.

[6] "MathWorks," [Online]. Available: http://www.mathworks.com/products/matlab/index.html.

[7] K. Waugh and J. M. Bach, "Biomechanics and Its Application to Seating," 2015.

APPENDIX A: MATLAB CODE

```
% Biomechanical Modeling of the Human Body for Application to Wheelchair
% Seating Systems

% Written by: Ryan Letcher
% Last edited: 3/31/2016

% Side view output of the body segment model

% This code requires inputs of overall body height and body weight of a
% patient as well as angles of certain joints for proper orientation of
the
% body in space. This code will output to the command window the forces
% from each support and will display a plot of the body segments and the
% point locations of the forces from each support.

clear all
close all
clc

%% Height of model
height = 69.1; %inches

%% Weight of model
weight = 172; %pounds

%% Body angle information
theta_leftshoulder_xy = 0;
theta_rightshoulder_xy = 0;
theta_leftelbow = 0;
theta_rightelbow = 0;
theta_leftwrist = 0;
theta_rightwrist = 0;
theta_leftpelvis = 100;
theta_rightpelvis = 100;
theta_leftknee = 90;
theta_rightknee = 90;
theta_leftankle = 0;
theta_rightankle = 0;
theta_trunk = 10;

%% Lengths, widths, and locations taken from "The Measure of Man and
Woman"
headlength = 8.7/69.1*height; %top to bottom and front to back
headwidth = 6.1/69.1*height; %left to right
shoulderwidth = 14.1/69.1*height; %left to right
upperarmlength = 11/69.1*height;
forearmlength = 10.1/69.1*height;
handlength = 7.5/69.1*height;
thighlength = (36.9-19.8)/69.1*height;
shanklength = (19.8-3.2)/69.1*height;
footlength = (10.4-2.5)/69.1*height; %heel to toe
footwidth = 3.9/69.1*height; %left to right
pelviswidth = 7/69.1*height; %left to right
```

```
necklength = (69.1-8.7-54.6)/69.1*height;
trunklength = (54.6-36.9)/69.1*height;
chestwidth = 12.2/69.1*height; %left to right

headlocation = (69.1-8.7)/69.1*height; %location of bottom of head
upperarmlocation = 54.6/69.1*height; %location of shoulder attachment
forearmlocation = (56.7-14.4+1.3)/69.1*height; %location of elbow
attachment
handlocation = forearmlocation-forearmlength; %location of wrist
attachment
shoulderlocation = shoulderwidth/2; %location along z-axis
anklelocation = 3.2/69.1*height; %location along y-axis
kneelocation = 19.8/69.1*height; %location along y-axis
thighlocation = 36.9/69.1*height; %location along y-axis
pelvislocation = pelviswidth/2; %location along z-axis

%% Segment weights taken from Winter's book as a percentage of the total
mass
weight_head = 8.1/100*weight;
weight_trunk = 49.7/100*weight;
weight_upperarm = 2.8/100*weight;
weight_forearm = 1.6/100*weight;
weight_hand = 0.6/100*weight;
weight_thigh = 10.0/100*weight;
weight_shank = 4.65/100*weight;
weight_foot = 1.45/100*weight;

%% CoG taken from Winter's book as a percentage of the length of each
segment
headcenter_xy = [-(trunklength+necklength+headlength/2)*sind(theta_trunk),
thighlocation+(trunklength+necklength+headlength/2)*cosd(theta_trunk)];
head = 0.598*headlength; %this taken from ASU document
headcog_xy = [-(trunklength+necklength+headlength-head)*sind(theta_trunk),
thighlocation+(trunklength+necklength+headlength-head)*cosd(theta_trunk)];
trunk = 0.500*trunklength;
trunkcog_xy = [-(trunklength-
trunk)*sind(theta_trunk),thighlocation+(trunklengthtrunk)*
cosd(theta_trunk)];
upperarm = 0.436*upperarmlength;
leftupperarmcog_xy = [upperarm*sind(theta_leftshoulder_xy+theta_trunk)-
trunklength*sind(theta_trunk),thighlocation+trunklength*cosd(theta_trunk)-
upperarm*cosd
(theta_leftshoulder_xy+theta_trunk)];
rightupperarmcog_xy = [upperarm*sind(theta_rightshoulder_xy+theta_trunk)-
trunklength*sind(theta_trunk),thighlocation+trunklength*cosd(theta_trunk)-
upperarm*cosd
(theta_rightshoulder_xy+theta_trunk)];
forearm = 0.430*forearmlength;
leftforearmcog_xy =
[upperarmlength*sind(theta_leftshoulder_xy+theta_trunk)-
trunklength*sind(theta_trunk) + forearm*sind
(theta_leftshoulder_xy+theta_leftelbow+theta_trunk),thighlocation+trunklength
*cosd
(theta_trunk)-upperarmlength*cosd(theta_leftshoulder_xy+theta_trunk) -
forearm*cosd
(theta_leftshoulder_xy+theta_leftelbow+theta_trunk)];
```

```
87 rightforearmcog_xy =
[upperarmlength*sind(theta_rightshoulder_xy+theta_trunk)-
trunklength*sind(theta_trunk) + forearm*sind
(theta_rightshoulder_xy+theta_rightelbow+theta_trunk),thighlocation+trunkleng
th*cosd
(theta_trunk)-upperarmlength*cosd(theta_rightshoulder_xy+theta_trunk) -
forearm*cosd
(theta_rightshoulder_xy+theta_rightelbow+theta_trunk)];
88 hand = 0.506*handlength;
89 lefthandcog_xy = [upperarmlength*sind(theta_leftshoulder_xy+theta_trunk)-
trunklength*sind(theta_trunk) + forearmlength*sind
(theta_leftshoulder_xy+theta_leftelbow+theta_trunk) + hand*sind
(theta_leftshoulder_xy+theta_leftelbow-theta_leftwrist+theta_trunk),
thighlocation+trunklength*cosd(theta_trunk)-upperarmlength*cosd
(theta_leftshoulder_xy+theta_trunk) - forearmlength*cosd
(theta_leftshoulder_xy+theta_leftelbow+theta_trunk) - hand*cosd
(theta_leftshoulder_xy+theta_leftelbow-theta_leftwrist+theta_trunk)];
90 righthandcog_xy =
[upperarmlength*sind(theta_rightshoulder_xy+theta_trunk)-
trunklength*sind(theta_trunk) + forearmlength*sind
(theta_rightshoulder_xy+theta_rightelbow+theta_trunk) + hand*sind
(theta_rightshoulder_xy+theta_rightelbow-theta_rightwrist+theta_trunk),
thighlocation+trunklength*cosd(theta_trunk)-upperarmlength*cosd
(theta_rightshoulder_xy+theta_trunk) - forearmlength*cosd
(theta_rightshoulder_xy+theta_rightelbow+theta_trunk) - hand*cosd
(theta_rightshoulder_xy+theta_rightelbow-theta_rightwrist+theta_trunk)];
91 thigh = 0.433*thighlength;
92 leftthighcog_xy = [thigh*sind(theta_leftpelvis),thighlocation - thigh*cosd
(theta_leftpelvis)];
93 rightthighcog_xy = [thigh*sind(theta_rightpelvis),thighlocation -
thigh*cosd
(theta_rightpelvis)];
94 shank = 0.433*shanklength;
95 leftshankcog_xy = [thighlength*sind(theta_leftpelvis) +
shank*sind(theta_leftpelvistheta_
leftknee),thighlocation - thighlength*cosd(theta_leftpelvis) - shank*cosd
(theta_leftpelvis-theta_leftknee)];
96 rightshankcog_xy = [thighlength*sind(theta_rightpelvis) + shank*sind
(theta_rightpelvis-theta_rightknee),thighlocation -
thighlength*cosd(theta_rightpelvis) -
shank*cosd(theta_rightpelvis-theta_rightknee)];
97 foot = 0.2654*footlength;
98 leftfootcog_xy = [thighlength*sind(theta_leftpelvis) + shanklength*sind
(theta_leftpelvis-theta_leftknee) + foot*sind(theta_leftpelvis-
theta_leftknee+
(90+theta_leftankle)) + 0.50*anklelocation*sind(theta_leftpelvistheta_
leftknee+theta_leftankle),thighlocation - thighlength*cosd(theta_leftpelvis)
-
shanklength*cosd(theta_leftpelvis-theta_leftknee) +
foot*cosd(theta_leftpelvistheta_
leftknee+(theta_leftankle-90)) -
0.50*anklelocation*cosd(theta_leftpelvistheta_
leftknee+theta_leftankle)];
99 rightfootcog_xy = [thighlength*sind(theta_rightpelvis) + shanklength*sind
(theta_rightpelvis-theta_rightknee) + foot*sind(theta_rightpelvis-
theta_rightknee+
```

```
(90+theta_rightankle)) + 0.50*anklelocation*sind(theta_rightpelvistheta_
rightknee+theta_rightankle),thighlocation -
thighlength*cosd(theta_rightpelvis) -
shanklength*cosd(theta_rightpelvis-theta_rightknee) +
foot*cosd(theta_rightpelvistheta_
rightknee+(theta_rightankle-90)) -
0.50*anklelocation*cosd(theta_rightpelvistheta_
rightknee+theta_rightankle)];
100
101 %% Whole body CoG in xy plane calculated from individual CoG's of body
segments
102 percenthead_xy = weight_head*headcog_xy;
103 percenttrunk_xy = weight_trunk*trunkcog_xy;
104 percentleftupperarm_xy = weight_upperarm*leftupperarmcog_xy;
105 percentrightupperarm_xy = weight_upperarm*rightupperarmcog_xy;
106 percentleftforearm_xy = weight_forearm*leftforearmcog_xy;
107 percentrightforearm_xy = weight_forearm*rightforearmcog_xy;
108 percentlefthand_xy = weight_hand*lefthandcog_xy;
109 percentrighthand_xy = weight_hand*righthandcog_xy;
110 percentleftthigh_xy = weight_thigh*leftthighcog_xy;
111 percentrightthigh_xy = weight_thigh*rightthighcog_xy;
112 percentleftshank_xy = weight_shank*leftshankcog_xy;
113 percentrightshank_xy = weight_shank*rightshankcog_xy;
114 percentleftfoot_xy = weight_foot*leftfootcog_xy;
115 percentrightfoot_xy = weight_foot*rightfootcog_xy;
116
117 bodycog_xy = (percenthead_xy + percenttrunk_xy + percentleftupperarm_xy +
percentrightupperarm_xy + percentleftforearm_xy + percentrightforearm_xy +
percentlefthand_xy + percentrighthand_xy + percentleftthigh_xy +
percentrightthigh_xy +
percentleftshank_xy + percentrightshank_xy + percentleftfoot_xy +
percentrightfoot_xy)
/weight;
118
119 %% Create plot of body segments in xy plane (side view)
120 trunk_x = [0 -(trunklength+necklength)*sind(theta_trunk)];
121 trunk_y = [thighlocation
thighlocation+(trunklength+necklength)*cosd(theta_trunk)];
122 leftupperarm_x = [-trunklength*sind(theta_trunk) -
trunklength*sind(theta_trunk)
+upperarmlength*sind(theta_leftshoulder_xy+theta_trunk)];
123 leftupperarm_y = [thighlocation+trunklength*cosd(theta_trunk)
thighlocation+trunklength*cosd(theta_trunk)-upperarmlength*cosd
(theta_leftshoulder_xy+theta_trunk)];
124 rightupperarm_x = [-trunklength*sind(theta_trunk) -
trunklength*sind(theta_trunk)
+upperarmlength*sind(theta_rightshoulder_xy+theta_trunk)];
125 rightupperarm_y = [thighlocation+trunklength*cosd(theta_trunk)
thighlocation+trunklength*cosd(theta_trunk)-upperarmlength*cosd
(theta_rightshoulder_xy+theta_trunk)];
126 leftforearm_x = [-trunklength*sind(theta_trunk)+upperarmlength*sind
(theta_leftshoulder_xy+theta_trunk) -
trunklength*sind(theta_trunk)+upperarmlength*sind
(theta_leftshoulder_xy+theta_trunk)+forearmlength*sind
(theta_leftelbow+theta_leftshoulder_xy+theta_trunk)];
```

```
127 leftforearm_y = [thighlocation+trunklength*cosd(theta_trunk)-
upperarmlength*cosd
(theta_leftshoulder_xy+theta_trunk)
thighlocation+trunklength*cosd(theta_trunk)-
upperarmlength*cosd(theta_leftshoulder_xy+theta_trunk)-forearmlength*cosd
(theta_leftelbow+theta_leftshoulder_xy+theta_trunk)];
128 rightforearm_x = [-trunklength*sind(theta_trunk)+upperarmlength*sind
(theta_rightshoulder_xy+theta_trunk) -
trunklength*sind(theta_trunk)+upperarmlength*sind
(theta_rightshoulder_xy+theta_trunk)+forearmlength*sind
(theta_rightelbow+theta_rightshoulder_xy+theta_trunk)];
129 rightforearm_y = [thighlocation+trunklength*cosd(theta_trunk)-
upperarmlength*cosd
(theta_rightshoulder_xy+theta_trunk)
thighlocation+trunklength*cosd(theta_trunk)-
upperarmlength*cosd(theta_rightshoulder_xy+theta_trunk)-forearmlength*cosd
(theta_rightelbow+theta_rightshoulder_xy+theta_trunk)];
130 lefthand_x = [-trunklength*sind(theta_trunk)+upperarmlength*sind
(theta_leftshoulder_xy+theta_trunk)+forearmlength*sind
(theta_leftelbow+theta_leftshoulder_xy+theta_trunk) -
trunklength*sind(theta_trunk)
+upperarmlength*sind(theta_leftshoulder_xy+theta_trunk)+forearmlength*sind
(theta_leftelbow+theta_leftshoulder_xy+theta_trunk)+handlength*sind(-
theta_leftwrist+theta_leftelbow+theta_leftshoulder_xy+theta_trunk)];
131 lefthand_y = [thighlocation+trunklength*cosd(theta_trunk)-
upperarmlength*cosd
(theta_leftshoulder_xy+theta_trunk)-forearmlength*cosd
(theta_leftelbow+theta_leftshoulder_xy+theta_trunk)
thighlocation+trunklength*cosd
(theta_trunk)-upperarmlength*cosd(theta_leftshoulder_xy+theta_trunk)-
forearmlength*cosd
(theta_leftelbow+theta_leftshoulder_xy+theta_trunk)-handlength*cosd(-
theta_leftwrist+theta_leftelbow+theta_leftshoulder_xy+theta_trunk)];
132 righthand_x = [-trunklength*sind(theta_trunk)+upperarmlength*sind
(theta_rightshoulder_xy+theta_trunk)+forearmlength*sind
(theta_rightelbow+theta_rightshoulder_xy+theta_trunk) -
trunklength*sind(theta_trunk)
+upperarmlength*sind(theta_rightshoulder_xy+theta_trunk)+forearmlength*sind
(theta_rightelbow+theta_rightshoulder_xy+theta_trunk)+handlength*sind(-
theta_rightwrist+theta_rightelbow+theta_rightshoulder_xy+theta_trunk)];
133 righthand_y = [thighlocation+trunklength*cosd(theta_trunk)-
upperarmlength*cosd
(theta_rightshoulder_xy+theta_trunk)-forearmlength*cosd
(theta_rightelbow+theta_rightshoulder_xy+theta_trunk)
thighlocation+trunklength*cosd
(theta_trunk)-upperarmlength*cosd(theta_rightshoulder_xy+theta_trunk)-
forearmlength*cosd
(theta_rightelbow+theta_rightshoulder_xy+theta_trunk)-handlength*cosd(-
theta_rightwrist+theta_rightelbow+theta_rightshoulder_xy+theta_trunk)];
134 leftthigh_x = [0 thighlength*sind(theta_leftpelvis)];
135 leftthigh_y = [thighlocation thighlocation -
thighlength*cosd(theta_leftpelvis)];
136 rightthigh_x = [0 thighlength*sind(theta_rightpelvis)];
137 rightthigh_y = [thighlocation thighlocation -
thighlength*cosd(theta_rightpelvis)];
```

```
138 leftshank_x = [thighlength*sind(theta_leftpelvis)
thighlength*sind(theta_leftpelvis)
+ shanklength*sind(theta_leftpelvis-theta_leftknee)];
139 leftshank_y = [thighlocation - thighlength*cosd(theta_leftpelvis)
thighlocation -
thighlength*cosd(theta_leftpelvis) - shanklength*cosd(theta_leftpelvis-
theta_leftknee)];
140 rightshank_x = [thighlength*sind(theta_rightpelvis) thighlength*sind
(theta_rightpelvis) + shanklength*sind(theta_rightpelvis-theta_rightknee)];
141 rightshank_y = [thighlocation - thighlength*cosd(theta_rightpelvis)
thighlocation -
thighlength*cosd(theta_rightpelvis) -
shanklength*cosd(theta_rightpelvistheta_
rightknee)];
142 leftankle_x = [thighlength*sind(theta_leftpelvis) + shanklength*sind
(theta_leftpelvis-theta_leftknee) thighlength*sind(theta_leftpelvis) +
shanklength*sind
(theta_leftpelvis-theta_leftknee) + anklelocation*sind(theta_leftpelvistheta_
leftknee+theta_leftankle)];
143 leftankle_y = [thighlocation - thighlength*cosd(theta_leftpelvis) -
shanklength*cosd
(theta_leftpelvis-theta_leftknee) thighlocation -
thighlength*cosd(theta_leftpelvis) -
shanklength*cosd(theta_leftpelvis-theta_leftknee) -
anklelocation*cosd(theta_leftpelvistheta_
leftknee+theta_leftankle)];
144 rightankle_x = [thighlength*sind(theta_rightpelvis) + shanklength*sind
(theta_rightpelvis-theta_rightknee) thighlength*sind(theta_rightpelvis) +
shanklength*sind(theta_rightpelvis-theta_rightknee) + anklelocation*sind
(theta_rightpelvis-theta_rightknee+theta_rightankle)];
145 rightankle_y = [thighlocation - thighlength*cosd(theta_rightpelvis) -
shanklength*cosd(theta_rightpelvis-theta_rightknee) thighlocation -
thighlength*cosd
(theta_rightpelvis) - shanklength*cosd(theta_rightpelvis-theta_rightknee) -
anklelocation*cosd(theta_rightpelvis-theta_rightknee+theta_rightankle)];
146 leftfoot_x = [thighlength*sind(theta_leftpelvis) +
shanklength*sind(theta_leftpelvistheta_
leftknee) + anklelocation*sind(theta_leftpelvis-
theta_leftknee+theta_leftankle)
thighlength*sind(theta_leftpelvis) + shanklength*sind(theta_leftpelvis-
theta_leftknee) +
footlength*sind(theta_leftpelvis-theta_leftknee+(90+theta_leftankle)) +
anklelocation*sind(theta_leftpelvis-theta_leftknee+theta_leftankle)];
147 leftfoot_y = [thighlocation - thighlength*cosd(theta_leftpelvis) -
shanklength*cosd
(theta_leftpelvis-theta_leftknee) - anklelocation*cosd(theta_leftpelvistheta_
leftknee+theta_leftankle) thighlocation - thighlength*cosd(theta_leftpelvis)
-
shanklength*cosd(theta_leftpelvis-theta_leftknee) +
footlength*cosd(theta_leftpelvistheta_
leftknee+(theta_leftankle-90)) - anklelocation*cosd(theta_leftpelvistheta_
leftknee+theta_leftankle)];
148 rightfoot_x = [thighlength*sind(theta_rightpelvis) + shanklength*sind
(theta_rightpelvis-theta_rightknee) +
anklelocation*sind(theta_rightpelvistheta_
```

```
rightknee+theta_rightankle) thighlength*sind(theta_rightpelvis) +
shanklength*sind
(theta_rightpelvis-theta_rightknee) + footlength*sind(theta_rightpelvis-
theta_rightknee+
(90+theta_rightankle)) + anklelocation*sind(theta_rightpelvistheta_
rightknee+theta_rightankle)];
149 rightfoot_y = [thighlocation - thighlength*cosd(theta_rightpelvis) -
shanklength*cosd
(theta_rightpelvis-theta_rightknee) -
anklelocation*cosd(theta_rightpelvistheta_
rightknee+theta_rightankle) thighlocation -
thighlength*cosd(theta_rightpelvis) -
shanklength*cosd(theta_rightpelvis-theta_rightknee) +
footlength*cosd(theta_rightpelvistheta_
rightknee+(theta_rightankle-90)) - anklelocation*cosd(theta_rightpelvistheta_
rightknee+theta_rightankle)];
150
151 plot1 = figure('DefaultAxesFontSize',20);
152 grid on;
153 line(trunk_x,trunk_y,'Color','k','LineWidth',5)
154 line(leftupperarm_x,leftupperarm_y,'Color',[0 .5 0],'LineWidth',5)
155 line(rightupperarm_x,rightupperarm_y,'Color','b','LineWidth',5)
156 line(leftforearm_x,leftforearm_y,'Color',[0 .5 0],'LineWidth',5)
157 line(rightforearm_x,rightforearm_y,'Color','b','LineWidth',5)
158 line(lefthand_x,lefthand_y,'Color',[0 .5 0],'LineWidth',5)
159 line(righthand_x,righthand_y,'Color','b','LineWidth',5)
160 line(leftthigh_x,leftthigh_y,'Color',[0 .5 0],'LineWidth',5)
161 line(rightthigh_x,rightthigh_y,'Color','b','LineWidth',5)
162 line(leftshank_x,leftshank_y,'Color',[0 .5 0],'LineWidth',5)
163 line(rightshank_x,rightshank_y,'Color','b','LineWidth',5)
164 line(leftankle_x,leftankle_y,'Color',[0 .5 0],'LineWidth',5)
165 line(rightankle_x,rightankle_y,'Color','b','LineWidth',5)
166 line(leftfoot_x,leftfoot_y,'Color',[0 .5 0],'LineWidth',5)
167 line(rightfoot_x,rightfoot_y,'Color','b','LineWidth',5)
168 axis([-(height+2)/2 (height+2)/2 0 height+2])
169 set(plot1,'Position',[200,100,1000,1000])
170 title('Human Body Segment Model: Side View','FontSize',24)
171 xlabel('x-axis (inches)','FontSize',24)
172 ylabel('y-axis (inches)','FontSize',24)
173 hold on
174 viscircles(headcenter_xy,headlength/2,'EdgeColor','k','LineWidth',5);
%assuming head is a circle
175
176 %Identify CoG locations in xy plane on plot
177 hold on
178 viscircles(headcog_xy,0.5,'LineWidth',5);
179 viscircles(trunkcog_xy,0.5,'LineWidth',5);
180 viscircles(leftupperarmcog_xy,0.5,'LineWidth',5);
181 viscircles(leftforearmcog_xy,0.5,'LineWidth',5);
182 viscircles(lefthandcog_xy,0.5,'LineWidth',5);
183 viscircles(leftthighcog_xy,0.5,'LineWidth',5);
184 viscircles(leftshankcog_xy,0.5,'LineWidth',5);
185 viscircles(leftfootcog_xy,0.5,'LineWidth',5);
186 viscircles(rightupperarmcog_xy,0.5,'LineWidth',5);
187 viscircles(rightforearmcog_xy,0.5,'LineWidth',5);
188 viscircles(righthandcog_xy,0.5,'LineWidth',5);
```

```
189 viscircles(rightthighcog_xy,0.5,'LineWidth',5);
190 viscircles(rightshankcog_xy,0.5,'LineWidth',5);
191 viscircles(rightfootcog_xy,0.5,'LineWidth',5);
192
193 %Identify CoG of whole body in xy plane on plot
194 hold on
195 viscircles(bodycog_xy,.75,'EdgeColor',[.5 0 .5],'LineWidth',8);
196
197 %% Calculate force of backrest
198
199 % Check to see if CoG of body segment is located over the backrest
200 back_total_force=0;
201 back_total_moment=0;
202 if headcog_xy(1)<0
203 back_total_force = back_total_force + weight_head;
204 back_total_moment = back_total_moment + headcog_xy(1)*weight_head;
205 end
206 if trunkcog_xy(1)<0
207 back_total_force = back_total_force + weight_trunk;
208 back_total_moment = back_total_moment + trunkcog_xy(1)*weight_trunk;
209 end
210 if leftupperarmcog_xy(1)<0
211 back_total_force = back_total_force + weight_upperarm;
212 back_total_moment = back_total_moment +
leftupperarmcog_xy(1)*weight_upperarm;
213 end
214 if rightupperarmcog_xy(1)<0
215 back_total_force = back_total_force + weight_upperarm;
216 back_total_moment = back_total_moment +
rightupperarmcog_xy(1)*weight_upperarm;
217 end
218 if leftforearmcog_xy(1)<0
219 back_total_force = back_total_force + weight_forearm;
220 back_total_moment = back_total_moment +
leftforearmcog_xy(1)*weight_forearm;
221 end
222 if rightforearmcog_xy(1)<0
223 back_total_force = back_total_force + weight_forearm;
224 back_total_moment = back_total_moment +
rightforearmcog_xy(1)*weight_forearm;
225 end
226 if lefthandcog_xy(1)<0
227 back_total_force = back_total_force + weight_hand;
228 back_total_moment = back_total_moment + lefthandcog_xy(1)*weight_hand;
229 end
230 if righthandcog_xy(1)<0
231 back_total_force = back_total_force + weight_hand;
232 back_total_moment = back_total_moment + righthandcog_xy(1)*weight_hand;
233 end
234
235 % Forces of backrest
236 force_backrest_normal = back_total_force*sind(theta_trunk)
237 force_backrest_friction = back_total_force*cosd(theta_trunk);
238 force_backrest_normal_x = force_backrest_normal*cosd(theta_trunk);
239 force_backrest_normal_y = force_backrest_normal*sind(theta_trunk);
240 force_backrest_friction_y = force_backrest_friction*cosd(theta_trunk);
```

```
241
242 % Location of point force of backrest
243 if force_backrest_normal>0 %if the backrest is used
244 forcepoint_backrest_length = -back_total_moment/force_backrest_normal;
245 forcepoint_backrest_x = -forcepoint_backrest_length*sind(theta_trunk);
246 forcepoint_backrest_y = thighlocation + forcepoint_backrest_length*cosd
(theta_trunk);
247viscircles([forcepoint_backrest_x,forcepoint_backrest_y],.75,'EdgeColor',[
.4 .4 .4],'LineWidth',8);
248 else %if the backrest is not used
249 forcepoint_backrest_length = 0;
250 forcepoint_backrest_x = 0;
251 forcepoint_backrest_y = 0;
252 end
253
254 %% Calculate force of cushion
255
256 % Location the cushion on the body
257 cushion_length = thighlength;
258 cushion_angle = min([theta_leftpelvis,theta_rightpelvis])-90; % relative
to horizontal
259 cushion_endpoint = cushion_length*cosd(cushion_angle);
260
261 % Check to see if CoG of body segment is located over the cushion
262 cushion_total_force=0;
263 cushion_total_moment=0;
264 if headcog_xy(1)>=0 && headcog_xy(1)<cushion_endpoint
265 cushion_total_force = cushion_total_force + weight_head;
266 cushion_total_moment = cushion_total_moment + headcog_xy(1)*weight_head;
267 end
268 if trunkcog_xy(1)>=0 && trunkcog_xy(1)<cushion_endpoint
269 cushion_total_force = cushion_total_force + weight_trunk;
270 cushion_total_moment = cushion_total_moment +
trunkcog_xy(1)*weight_trunk;
271 end
272 if leftupperarmcog_xy(1)>=0 && leftupperarmcog_xy(1)<cushion_endpoint
273 cushion_total_force = cushion_total_force + weight_upperarm;
274 cushion_total_moment = cushion_total_moment + leftupperarmcog_xy(1)
*weight_upperarm;
275 end
276 if rightupperarmcog_xy(1)>=0 && rightupperarmcog_xy(1)<cushion_endpoint
277 cushion_total_force = cushion_total_force + weight_upperarm;
278 cushion_total_moment = cushion_total_moment + rightupperarmcog_xy(1)
*weight_upperarm;
279 end
280 if leftforearmcog_xy(1)>=0 && leftforearmcog_xy(1)<cushion_endpoint
281 cushion_total_force = cushion_total_force + weight_forearm;
282 cushion_total_moment = cushion_total_moment + leftforearmcog_xy(1)
*weight_forearm;
283 end
284 if rightforearmcog_xy(1)>=0 && rightforearmcog_xy(1)<cushion_endpoint
285 cushion_total_force = cushion_total_force + weight_forearm;
286 cushion_total_moment = cushion_total_moment + rightforearmcog_xy(1)
*weight_forearm;
287 end
288 if lefthandcog_xy(1)>=0 && lefthandcog_xy(1)<cushion_endpoint
```

```
289 cushion_total_force = cushion_total_force + weight_hand;
290 cushion_total_moment = cushion_total_moment +
lefthandcog_xy(1)*weight_hand;
291 end
292 if righthandcog_xy(1)>=0 && righthandcog_xy(1)<cushion_endpoint
293 cushion_total_force = cushion_total_force + weight_hand;
294 cushion_total_moment = cushion_total_moment +
righthandcog_xy(1)*weight_hand;
295 end
296 if leftthighcog_xy(1)>=0 && leftthighcog_xy(1)<cushion_endpoint
297 cushion_total_force = cushion_total_force + weight_thigh;
298 cushion_total_moment = cushion_total_moment +
leftthighcog_xy(1)*weight_thigh;
299 end
300 if rightthighcog_xy(1)>=0 && rightthighcog_xy(1)<cushion_endpoint
301 cushion_total_force = cushion_total_force + weight_thigh;
302 cushion_total_moment = cushion_total_moment +
rightthighcog_xy(1)*weight_thigh;
303 end
304
305 % Forces of cushion
306 force_cushion_normal = (cushion_total_force +
force_backrest_friction_y)*cosd(cushion_angle) +
force_backrest_normal_x*sind(cushion_angle)
307 force_cushion_friction = (cushion_total_force +
force_backrest_friction_y)*sind
(cushion_angle) - force_backrest_normal_x*cosd(cushion_angle)
308 force_cushion_normal_x = -force_cushion_normal*sind(cushion_angle);
309 force_cushion_normal_y = force_cushion_normal*cosd(cushion_angle);
310 force_cushion_friction_x = force_cushion_friction*cosd(cushion_angle);
311 force_cushion_friction_y = force_cushion_friction*sind(cushion_angle);
312 force_cushion = sqrt(force_cushion_normal^2 + force_cushion_friction^2);
313
314 % Location of point force of cushion
315 forcepoint_cushion_length = cushion_total_moment/force_cushion_normal;
316 forcepoint_cushion_x = forcepoint_cushion_length*cosd(cushion_angle);
317 forcepoint_cushion_y = thighlocation +
forcepoint_cushion_length*sind(cushion_angle);
318 viscircles([forcepoint_cushion_x,forcepoint_cushion_y],.75,'EdgeColor',[0
.50],'LineWidth',8);
319
320 % Assume no friction on backrest
321 force_backrest_friction = 0;
322 force_backrest_friction_x = 0;
323 force_backrest_friction_y = 0;
324
325 %% Calculate force of footrests
326
327 % Location of footrests on the body
328 footrest_angle = min([theta_leftpelvis-theta_leftknee+theta_leftankle,
theta_rightpelvis-theta_rightknee+theta_rightankle]);
329
330 % Check to see if CoG of body segment is located over the footrests
331 footrest_total_force=0;
332 footrest_total_moment=0;
333 if leftupperarmcog_xy(1)>=cushion_endpoint
```

```
334 footrest_total_force = footrest_total_force + weight_upperarm;
335 footrest_total_moment = footrest_total_moment + (leftupperarmcog_xy(1)-
cushion_endpoint)*weight_upperarm;
336 end
337 if rightupperarmcog_xy(1)>=cushion_endpoint
338 footrest_total_force = footrest_total_force + weight_upperarm;
339 footrest_total_moment = footrest_total_moment + (rightupperarmcog_xy(1)-
cushion_endpoint)*weight_upperarm;
340 end
341 if leftforearmcog_xy(1)>=cushion_endpoint
342 footrest_total_force = footrest_total_force + weight_forearm;
343 footrest_total_moment = footrest_total_moment + (leftforearmcog_xy(1)-
cushion_endpoint)*weight_forearm;
344 end
345 if rightforearmcog_xy(1)>=cushion_endpoint
346 footrest_total_force = footrest_total_force + weight_forearm;
347 footrest_total_moment = footrest_total_moment + (rightforearmcog_xy(1)-
cushion_endpoint)*weight_forearm;
348 end
349 if lefthandcog_xy(1)>=cushion_endpoint
350 footrest_total_force = footrest_total_force + weight_hand;
351 footrest_total_moment = footrest_total_moment + (lefthandcog_xy(1)-
cushion_endpoint)*weight_hand;
352 end
353 if righthandcog_xy(1)>=cushion_endpoint
354 footrest_total_force = footrest_total_force + weight_hand;
355 footrest_total_moment = footrest_total_moment + (righthandcog_xy(1)-
cushion_endpoint)*weight_hand;
356 end
357 footrest_total_force = footrest_total_force + weight_shank;
358 footrest_total_moment = footrest_total_moment + (leftshankcog_xy(1)-
cushion_endpoint)
*weight_shank;
359 footrest_total_force = footrest_total_force + weight_shank;
360 footrest_total_moment = footrest_total_moment + (rightshankcog_xy(1)-
cushion_endpoint)*weight_shank;
361 footrest_total_force = footrest_total_force + weight_foot;
362 footrest_total_moment = footrest_total_moment + (leftfootcog_xy(1)-
cushion_endpoint)
*weight_foot;
363 footrest_total_force = footrest_total_force + weight_foot;
364 footrest_total_moment = footrest_total_moment + (rightfootcog_xy(1)-
cushion_endpoint)*weight_foot;
365
366 % Forces of footrests
367 force_footrest_normal = footrest_total_force*cosd(footrest_angle)
368 force_footrest_friction = footrest_total_force*sind(footrest_angle)
369 force_footrest_normal_x = -force_footrest_normal*sind(footrest_angle);
370 force_footrest_normal_y = force_footrest_normal*cosd(footrest_angle);
371 force_footrest_friction_x = force_footrest_friction*cosd(footrest_angle);
372 force_footrest_friction_y = force_footrest_friction*sind(footrest_angle);
373
374 % Locations of point forces of footrests
375 forcepoint_footrest = footrest_total_moment/force_footrest_friction;
376 forcepoint_footrest1_x = cushion_endpoint +
(shanklength+anklelocation)*sind
```

```
(footrest_angle);
377 forcepoint_footrest1_y = thighlocation +
cushion_length*sind(cushion_angle) -
(shanklength+anklelocation)*cosd(footrest_angle);
378 forcepoint_footrest2_x = cushion_endpoint +
forcepoint_footrest*sind(footrest_angle);
379 forcepoint_footrest2_y = thighlocation +
cushion_length*sind(cushion_angle) -
forcepoint_footrest*cosd(footrest_angle);
380
viscircles([forcepoint_footrest1_x,forcepoint_footrest1_y],.75,'EdgeColor',[.
5 .4 0],'LineWidth',8);
381
viscircles([forcepoint_footrest2_x,forcepoint_footrest2_y],.75,'EdgeColor',[.
5 .4 0],'LineWidth',8);
382
```

```
% Biomechanical Modeling of the Human Body for Application to Wheelchair
% Seating Systems

% Written by: Ryan Letcher
% Last edited: 3/31/2016

% Front view output of the body segment model

% This code requires inputs of overall body height and body weight of a
% patient as well as angles of certain joints for proper orientation of
the
% body in space. This code will output to the command window the forces
% from each support and will display a plot of the body segments and the
% point locations of the forces from each support.

clear all
close all
clc

%% Height of model
height = 69.1; %inches

%% Weight of model
weight = 172; %pounds

%% Body angle information
theta_leftshoulder_yz = 0;
theta_rightshoulder_yz = 0;
theta_leftpelvis = 90;
theta_rightpelvis = 90;
theta_leftknee = 90;
theta_rightknee = 90;
theta_leftankle = 0;
theta_rightankle = 0;
theta_trunk_yz = 0;
left_lateral_distance = 12; %distance from pelvis to center of lateral
support
right_lateral_distance = 12; %distance from pelvis to center of lateral
support

%% Lengths, widths, and locations taken from "The Measure of Man and
Woman"
headlength = 8.7/69.1*height; %top to bottom and front to back
headwidth = 6.1/69.1*height; %left to right
shoulderwidth = 14.1/69.1*height; %left to right
upperarmlength = 11/69.1*height;
forearmlength = 10.1/69.1*height;
handlength = 7.5/69.1*height;
thighlength = (36.9-19.8)/69.1*height;
shanklength = (19.8-3.2)/69.1*height;
footlength = (10.4-2.5)/69.1*height; %heel to toe
footwidth = 3.9/69.1*height; %left to right
pelviswidth = 7/69.1*height; %left to right
necklength = (69.1-8.7-54.6)/69.1*height;
trunklength = (54.6-36.9)/69.1*height;
chestwidth = 12.2/69.1*height; %left to right
```

```
53
54 headlocation = (69.1-8.7)/69.1*height; %location of bottom of head
55 upperarmlocation = 54.6/69.1*height; %location of shoulder attachment
56 forearmlocation = (56.7-14.4+1.3)/69.1*height; %location of elbow
attachment
57 handlocation = forearmlocation-forearmlength; %location of wrist
attachment
58 shoulderlocation = shoulderwidth/2; %location along z-axis
59 anklelocation = 3.2/69.1*height; %location along y-axis
60 kneelocation = 19.8/69.1*height; %location along y-axis
61 thighlocation = 36.9/69.1*height; %location along y-axis
62 pelvislocation = pelviswidth/2; %location along z-axis
63
64 %% Segment weights taken from Winter's book as a percentage of the total
mass
65 weight_head = 8.1/100*weight;
66 weight_trunk = 49.7/100*weight;
67 weight_upperarm = 2.8/100*weight;
68 weight_forearm = 1.6/100*weight;
69 weight_hand = 0.6/100*weight;
70 weight_thigh = 10.0/100*weight;
71 weight_shank = 4.65/100*weight;
72 weight_foot = 1.45/100*weight;
73
74 %% CoG taken from Winter's book as a percentage of the length of each
segment
75 headcenter_yz =
[(trunklength+necklength+headlength/2)*sind(theta_trunk_yz),
thighlocation+(trunklength+necklength+headlength/2)*cosd(theta_trunk_yz)];
76 head = 0.598*headlength; %this taken from ASU document
77 headcog_yz = [(trunklength+necklength+headlength-
head)*sind(theta_trunk_yz),
thighlocation+(trunklength+necklength+headlength-head)*cosd(theta_trunk_yz)];
78 trunk = 0.500*trunklength;
79 trunkcog_yz = [(trunklength-
trunk)*sind(theta_trunk_yz),thighlocation+(trunklengthtrunk)*
cosd(theta_trunk_yz)];
80 upperarm = 0.436*upperarmlength;
81 pelvis_to_shoulder_length = sqrt(shoulderlocation^2+(upperarmlocation-
thighlocation)^2);
82 pelvis_to_shoulder_angle = atand(shoulderlocation/(upperarmlocation-
thighlocation));
83 leftupperarmcog_yz = [pelvis_to_shoulder_length*sind
(pelvis_to_shoulder_angle+theta_trunk_yz)+upperarm*sind(theta_leftshoulder_yz
theta_trunk_yz),thighlocation+pelvis_to_shoulder_length*cosd
(pelvis_to_shoulder_angle+theta_trunk_yz)-
upperarm*cosd(theta_leftshoulder_yztheta_trunk_yz)];
84 rightupperarmcog_yz = [-
pelvis_to_shoulder_length*sind(pelvis_to_shoulder_angletheta_
trunk_yz)-upperarm*sind(theta_rightshoulder_yz+theta_trunk_yz),
thighlocation+pelvis_to_shoulder_length*cosd(pelvis_to_shoulder_angle-
theta_trunk_yz)-
upperarm*cosd(theta_rightshoulder_yz+theta_trunk_yz)];
85 forearm = 0.430*forearmlength;
86 leftforearmcog_yz = [pelvis_to_shoulder_length*sind
(pelvis_to_shoulder_angle+theta_trunk_yz)+(upperarmlength+forearm)*sind
```

```
(theta_leftshoulder_yz-
theta_trunk_yz),thighlocation+pelvis_to_shoulder_length*cosd
(pelvis_to_shoulder_angle+theta_trunk_yz)-(upperarmlength+forearm)*cosd
(theta_leftshoulder_yz-theta_trunk_yz)];
87 rightforearmcog_yz = [-
pelvis_to_shoulder_length*sind(pelvis_to_shoulder_angle-theta_trunk_yz)-
(upperarmlength+forearm)*sind(theta_rightshoulder_yz+theta_trunk_yz),
thighlocation+pelvis_to_shoulder_length*cosd(pelvis_to_shoulder_angle-
theta_trunk_yz)-
(upperarmlength+forearm)*cosd(theta_rightshoulder_yz+theta_trunk_yz)];
88 hand = 0.506*handlength;
89 lefthandcog_yz = [pelvis_to_shoulder_length*sind
(pelvis_to_shoulder_angle+theta_trunk_yz)+(upperarmlength+forearmlength+hand)
*sind(theta_leftshoulder_yz-
theta_trunk_yz),thighlocation+pelvis_to_shoulder_length*cosd
(pelvis_to_shoulder_angle+theta_trunk_yz)-
(upperarmlength+forearmlength+hand)*cosd
(theta_leftshoulder_yz-theta_trunk_yz)];
90 righthandcog_yz = [-
pelvis_to_shoulder_length*sind(pelvis_to_shoulder_angletheta_
trunk_yz)-(upperarmlength+forearmlength+hand)*sind
(theta_rightshoulder_yz+theta_trunk_yz),thighlocation+pelvis_to_shoulder_leng
th*cosd
(pelvis_to_shoulder_angle-theta_trunk_yz)-
(upperarmlength+forearmlength+hand)*cosd
(theta_rightshoulder_yz+theta_trunk_yz)];
91 thigh = 0.433*thighlength;
92 leftthighcog_yz = [pelvislocation,thighlocation-
thigh*cosd(theta_leftpelvis)];
93 rightthighcog_yz = [-pelvislocation,thighlocation-
thigh*cosd(theta_rightpelvis)];
94 shank = 0.433*shanklength;
95 leftshankcog_yz = [pelvislocation,thighlocation-
thighlength*cosd(theta_leftpelvis)-
shank*cosd(theta_leftpelvis-theta_leftknee)];
96 rightshankcog_yz = [-pelvislocation,thighlocation-
thighlength*cosd(theta_rightpelvis)-shank*cosd(theta_rightpelvis-
theta_rightknee)];
97 foot = 0.2654*footlength;
98 leftfootcog_yz = [pelvislocation,thighlocation-
thighlength*cosd(theta_leftpelvis)-shanklength*cosd(theta_leftpelvis-
theta_leftknee)+foot*cosd(theta_leftpelvistheta_leftknee+(theta_leftankle-
90))-
0.50*anklelocation*cosd(theta_leftpelvistheta_leftknee+theta_leftankle)];
99 rightfootcog_yz = [-pelvislocation,thighlocation-
thighlength*cosd(theta_rightpelvis)-shanklength*cosd(theta_rightpelvis-
theta_rightknee)+foot*cosd(theta_rightpelvistheta_
rightknee+(theta_rightankle-90))-
0.50*anklelocation*cosd(theta_rightpelvistheta_
rightknee+theta_rightankle)];
100
101 %% Whole body CoG in yz plane calculated from individual CoG's of body
segments
102 percenthead_yz = weight_head*headcog_yz;
103 percenttrunk_yz = weight_trunk*trunkcog_yz;
104 percentleftupperarm_yz = weight_upperarm*leftupperarmcog_yz;
```

```
105 percentrightupperarm_yz = weight_upperarm*rightupperarmcog_yz;
106 percentleftforearm_yz = weight_forearm*leftforearmcog_yz;
107 percentrightforearm_yz = weight_forearm*rightforearmcog_yz;
108 percentlefthand_yz = weight_hand*lefthandcog_yz;
109 percentrighthand_yz = weight_hand*righthandcog_yz;
110 percentleftthigh_yz = weight_thigh*leftthighcog_yz;
111 percentrightthigh_yz = weight_thigh*rightthighcog_yz;
112 percentleftshank_yz = weight_shank*leftshankcog_yz;
113 percentrightshank_yz = weight_shank*rightshankcog_yz;
114 percentleftfoot_yz = weight_foot*leftfootcog_yz;
115 percentrightfoot_yz = weight_foot*rightfootcog_yz;
116
117 bodycog_yz = (percenthead_yz + percenttrunk_yz + percentleftupperarm_yz +
percentrightupperarm_yz + percentleftforearm_yz + percentrightforearm_yz +
percentlefthand_yz + percentrighthand_yz + percentleftthigh_yz +
percentrightthigh_yz +
percentleftshank_yz + percentrightshank_yz + percentleftfoot_yz +
percentrightfoot_yz)
/weight;
118
119 %% Create plot of body segments in yz plane (front view)
120 trunk_z = [0 (trunklength+necklength)*sind(theta_trunk_yz)];
121 trunk_front_y = [thighlocation
thighlocation+(trunklength+necklength)*cosd(theta_trunk_yz)];
122 shoulders_z = [-
pelvis_to_shoulder_length*sind(pelvis_to_shoulder_angletheta_trunk_yz)
pelvis_to_shoulder_length*sind(pelvis_to_shoulder_angle+theta_trunk_yz)];
123 shoulders_y =
[thighlocation+pelvis_to_shoulder_length*cosd(pelvis_to_shoulder_angletheta_
trunk_yz) thighlocation+pelvis_to_shoulder_length*cosd
(pelvis_to_shoulder_angle+theta_trunk_yz)];
124 leftupperarm_z = [pelvis_to_shoulder_length*sind
(pelvis_to_shoulder_angle+theta_trunk_yz) pelvis_to_shoulder_length*sind
(pelvis_to_shoulder_angle+theta_trunk_yz)+upperarmlength*sind(theta_leftshoul
der_yztheta_trunk_yz)];
125 leftupperarm_front_y = [thighlocation+pelvis_to_shoulder_length*cosd
(pelvis_to_shoulder_angle+theta_trunk_yz)
thighlocation+pelvis_to_shoulder_length*cosd
(pelvis_to_shoulder_angle+theta_trunk_yz)-
upperarmlength*cosd(theta_leftshoulder_yztheta_trunk_yz)];
126 rightupperarm_z = [-
pelvis_to_shoulder_length*sind(pelvis_to_shoulder_angletheta_
trunk_yz) -pelvis_to_shoulder_length*sind(pelvis_to_shoulder_angle-
theta_trunk_yz)-
upperarmlength*sind(theta_rightshoulder_yz+theta_trunk_yz)];
127 rightupperarm_front_y = [thighlocation+pelvis_to_shoulder_length*cosd
(pelvis_to_shoulder_angle-theta_trunk_yz)
thighlocation+pelvis_to_shoulder_length*cosd
(pelvis_to_shoulder_angle-theta_trunk_yz)-upperarmlength*cosd
(theta_rightshoulder_yz+theta_trunk_yz)];
128 leftforearm_z = [pelvis_to_shoulder_length*sind
(pelvis_to_shoulder_angle+theta_trunk_yz)+upperarmlength*sind(theta_leftshoul
der_yztheta_trunk_yz)
pelvis_to_shoulder_length*sind(pelvis_to_shoulder_angle+theta_trunk_yz)+
(upperarmlength+forearmlength)*sind(theta_leftshoulder_yz-theta_trunk_yz)];
129 leftforearm_front_y = [thighlocation+pelvis_to_shoulder_length*cosd
```

```
(pelvis_to_shoulder_angle+theta_trunk_yz)-
upperarmlength*cosd(theta_leftshoulder_yztheta_
trunk_yz) thighlocation+pelvis_to_shoulder_length*cosd
(pelvis_to_shoulder_angle+theta_trunk_yz)-(upperarmlength+forearmlength)*cosd
(theta_leftshoulder_yz-theta_trunk_yz)];
130 rightforearm_z = [-
pelvis_to_shoulder_length*sind(pelvis_to_shoulder_angletheta_
trunk_yz)-upperarmlength*sind(theta_rightshoulder_yz+theta_trunk_yz) -
pelvis_to_shoulder_length*sind(pelvis_to_shoulder_angle-theta_trunk_yz)-
(upperarmlength+forearmlength)*sind(theta_rightshoulder_yz+theta_trunk_yz)];
131 rightforearm_front_y = [thighlocation+pelvis_to_shoulder_length*cosd
(pelvis_to_shoulder_angle-theta_trunk_yz)-upperarmlength*cosd
(theta_rightshoulder_yz+theta_trunk_yz)
thighlocation+pelvis_to_shoulder_length*cosd
(pelvis_to_shoulder_angle-theta_trunk_yz)-(upperarmlength+forearmlength)*cosd
(theta_rightshoulder_yz+theta_trunk_yz)];
132 lefthand_z =
[pelvis_to_shoulder_length*sind(pelvis_to_shoulder_angle+theta_trunk_yz)
+(upperarmlength+forearmlength)*sind(theta_leftshoulder_yz-theta_trunk_yz)
pelvis_to_shoulder_length*sind(pelvis_to_shoulder_angle+theta_trunk_yz)+
(upperarmlength+forearmlength+handlength)*sind(theta_leftshoulder_yz-
theta_trunk_yz)];
133 lefthand_front_y = [thighlocation+pelvis_to_shoulder_length*cosd
(pelvis_to_shoulder_angle+theta_trunk_yz)-(upperarmlength+forearmlength)*cosd
(theta_leftshoulder_yz-theta_trunk_yz)
thighlocation+pelvis_to_shoulder_length*cosd
(pelvis_to_shoulder_angle+theta_trunk_yz)-
(upperarmlength+forearmlength+handlength)*cosd
(theta_leftshoulder_yz-theta_trunk_yz)];
134 righthand_z = [-
pelvis_to_shoulder_length*sind(pelvis_to_shoulder_angletheta_
trunk_yz)-(upperarmlength+forearmlength)*sind
(theta_rightshoulder_yz+theta_trunk_yz) -pelvis_to_shoulder_length*sind
(pelvis_to_shoulder_angle-theta_trunk_yz)-
(upperarmlength+forearmlength+handlength)*sind
(theta_rightshoulder_yz+theta_trunk_yz)];
135 righthand_front_y = [thighlocation+pelvis_to_shoulder_length*cosd
(pelvis_to_shoulder_angle-theta_trunk_yz)-(upperarmlength+forearmlength)*cosd
(theta_rightshoulder_yz+theta_trunk_yz)
thighlocation+pelvis_to_shoulder_length*cosd
(pelvis_to_shoulder_angle-theta_trunk_yz)-
(upperarmlength+forearmlength+handlength)*cosd
(theta_rightshoulder_yz+theta_trunk_yz)];
136 pelvis_z = [-pelvislocation pelvislocation];
137 pelvis_y = [thighlocation thighlocation];
138 leftthigh_z = [pelvislocation pelvislocation];
139 leftthigh_front_y = [thighlocation thighlocation-
thighlength*cosd(theta_leftpelvis)];
140 rightthigh_z = [-pelvislocation -pelvislocation];
141 rightthigh_front_y = [thighlocation thighlocation-thighlength*cosd
(theta_rightpelvis)];
142 leftshank_z = [pelvislocation pelvislocation];
143 leftshank_front_y = [thighlocation-thighlength*cosd(theta_leftpelvis)
thighlocationthighlength*
cosd(theta_leftpelvis)-shanklength*cosd(theta_leftpelvis-theta_leftknee)];
144 rightshank_z = [-pelvislocation -pelvislocation];
```

```
145 rightshank_front_y = [thighlocation-thighlength*cosd(theta_rightpelvis)
thighlocation-thighlength*cosd(theta_rightpelvis)-
shanklength*cosd(theta_rightpelvistheta_
rightknee)];
146 leftankle_z = [pelvislocation pelvislocation];
147 leftankle_front_y = [thighlocation-thighlength*cosd(theta_leftpelvis)-
shanklength*cosd(theta_leftpelvis-theta_leftknee) thighlocation-
thighlength*cosd
(theta_leftpelvis)-shanklength*cosd(theta_leftpelvis-theta_leftknee)-
anklelocation*cosd
(theta_leftpelvis-theta_leftknee+theta_leftankle)];
148 rightankle_z = [-pelvislocation -pelvislocation];
149 rightankle_front_y = [thighlocation-thighlength*cosd(theta_rightpelvis)-
shanklength*cosd(theta_rightpelvis-theta_rightknee) thighlocation-
thighlength*cosd
(theta_rightpelvis)-shanklength*cosd(theta_rightpelvis-theta_rightknee)-
anklelocation*cosd(theta_rightpelvis-theta_rightknee+theta_rightankle)];
150 leftfoot_z = [pelvislocation-footwidth/2 pelvislocation+footwidth/2];
151 leftfoot_front_y = [thighlocation-thighlength*cosd(theta_leftpelvis)-
shanklength*cosd
(theta_leftpelvis-theta_leftknee)-anklelocation*cosd(theta_leftpelvistheta_
leftknee+theta_leftankle) thighlocation-thighlength*cosd(theta_leftpelvis)-
shanklength*cosd(theta_leftpelvis-theta_leftknee)-
anklelocation*cosd(theta_leftpelvistheta_
leftknee+theta_leftankle)];
152 rightfoot_z = [-pelvislocation-footwidth/2 -pelvislocation+footwidth/2];
153 rightfoot_front_y = [thighlocation-thighlength*cosd(theta_rightpelvis)-
shanklength*cosd(theta_rightpelvis-theta_rightknee)-
anklelocation*cosd(theta_rightpelvistheta_
rightknee+theta_rightankle) thighlocation-
thighlength*cosd(theta_rightpelvis)-
shanklength*cosd(theta_rightpelvis-theta_rightknee)-
anklelocation*cosd(theta_rightpelvistheta_
rightknee+theta_rightankle)];
154
155 plot1 = figure('DefaultAxesFontSize',20);
156 grid on;
157 line(trunk_z,trunk_front_y,'Color','k','LineWidth',5)
158 line(shoulders_z,shoulders_y,'Color','k','LineWidth',5)
159 line(pelvis_z,pelvis_y,'Color','k','LineWidth',5)
160 line(leftupperarm_z,leftupperarm_front_y,'Color',[0 .5 0],'LineWidth',5)
161 line(rightupperarm_z,rightupperarm_front_y,'Color','b','LineWidth',5)
162 line(leftforearm_z,leftforearm_front_y,'Color',[0 .5 0],'LineWidth',5)
163 line(rightforearm_z,rightforearm_front_y,'Color','b','LineWidth',5)
164 line(lefthand_z,lefthand_front_y,'Color',[0 .5 0],'LineWidth',5)
165 line(righthand_z,righthand_front_y,'Color','b','LineWidth',5)
166 line(leftthigh_z,leftthigh_front_y,'Color',[0 .5 0],'LineWidth',5)
167 line(rightthigh_z,rightthigh_front_y,'Color','b','LineWidth',5)
168 line(leftshank_z,leftshank_front_y,'Color',[0 .5 0],'LineWidth',5)
169 line(rightshank_z,rightshank_front_y,'Color','b','LineWidth',5)
170 line(leftankle_z,leftankle_front_y,'Color',[0 .5 0],'LineWidth',5)
171 line(rightankle_z,rightankle_front_y,'Color','b','LineWidth',5)
172 line(leftfoot_z,leftfoot_front_y,'Color',[0 .5 0],'LineWidth',5)
173 line(rightfoot_z,rightfoot_front_y,'Color','b','LineWidth',5)
174 axis([-36 36 0 72])
175 set(plot1,'Position',[200,100,1000,1000])
```

```
176 title('Human Body Segment Model: Front View','FontSize',24)
177 xlabel('z-axis (inches)','FontSize',24)
178 ylabel('y-axis (inches)','FontSize',24)
179 hold on
180 viscircles(headcenter_yz,headlength/2,'EdgeColor','k','LineWidth',5);
%assuming head is a circle
181
182 %Identify CoG locations in yz plane on plot
183 hold on
184 viscircles(headcog_yz,0.5,'LineWidth',5);
185 viscircles(trunkcog_yz,0.5,'LineWidth',5);
186 viscircles(leftupperarmcog_yz,0.5,'LineWidth',5);
187 viscircles(leftforearmcog_yz,0.5,'LineWidth',5);
188 viscircles(lefthandcog_yz,0.5,'LineWidth',5);
189 viscircles(leftthighcog_yz,0.5,'LineWidth',5);
190 viscircles(leftshankcog_yz,0.5,'LineWidth',5);
191 viscircles(leftfootcog_yz,0.5,'LineWidth',5);
192 viscircles(rightupperarmcog_yz,0.5,'LineWidth',5);
193 viscircles(rightforearmcog_yz,0.5,'LineWidth',5);
194 viscircles(righthandcog_yz,0.5,'LineWidth',5);
195 viscircles(rightthighcog_yz,0.5,'LineWidth',5);
196 viscircles(rightshankcog_yz,0.5,'LineWidth',5);
197 viscircles(rightfootcog_yz,0.5,'LineWidth',5);
198
199 %Identify CoG of whole body in yz plane on plot
200 hold on
201 viscircles(bodycog_yz,0.75,'EdgeColor',[0.5 0 0.5],'LineWidth',8);
202
203 %% Calculate point forces of cushion, laterals, and footrests
204 upperbodyandthighweight = weight_head+weight_trunk+2*
(weight_upperarm+weight_forearm+weight_hand+weight_thigh);
205 upperbodyweight = weight_head+weight_trunk+2*
(weight_upperarm+weight_forearm+weight_hand);
206 upperbodycog_yz = (percenthead_yz + percenttrunk_yz +
percentleftupperarm_yz +
percentrightupperarm_yz + percentleftforearm_yz + percentrightforearm_yz +
percentlefthand_yz + percentrighthand_yz)/upperbodyweight;
207 lowerbodyweight = 2*(weight_shank+weight_foot);
208
209 % Force on footrests
210 force_left_footrest = lowerbodyweight/2
211 force_right_footrest = lowerbodyweight/2
212
213 % Location of force on footrests
214 forcepoint_left_footrest_z = pelvislocation;
215 forcepoint_left_footrest_y = thighlocation-
thighlength*cosd(theta_leftpelvis)-
shanklength*cosd(theta_leftpelvis-theta_leftknee)-
anklelocation*cosd(theta_leftpelvistheta_
leftknee+theta_leftankle);
216 forcepoint_right_footrest_z = -pelvislocation;
217 forcepoint_right_footrest_y = thighlocation-
thighlength*cosd(theta_rightpelvis)-
shanklength*cosd(theta_rightpelvis-theta_rightknee)-
anklelocation*cosd(theta_rightpelvistheta_
rightknee+theta_rightankle);
```

```
218
viscircles([forcepoint_left_footrest_z,forcepoint_left_footrest_y],.75,'EdgeC
olor',[.5 .4 0],'LineWidth',8);
219
viscircles([forcepoint_right_footrest_z,forcepoint_right_footrest_y],.75,'Edg
eColor',[.5 .4 0],'LineWidth',8);
220
221 % Force on laterals
222 if theta_trunk_yz > 0
223 force_left_lateral = upperbodyweight*sind(theta_trunk_yz)
224 force_right_lateral = 0
225 elseif theta_trunk_yz < 0
226 force_right_lateral = upperbodyweight*sind(-theta_trunk_yz)
227 force_left_lateral = 0
228 else
229 force_left_lateral = 0
230 force_right_lateral = 0
231 end
232
233 % Location of force on laterals
234 pelvis_to_leftchest_length =
sqrt((chestwidth/2)^2+left_lateral_distance^2);
235 pelvis_to_leftchest_angle = atand((chestwidth/2)/left_lateral_distance);
236 pelvis_to_rightchest_length =
sqrt((chestwidth/2)^2+right_lateral_distance^2);
237 pelvis_to_rightchest_angle =
atand((chestwidth/2)/right_lateral_distance);
238 if theta_trunk_yz > 0
239 forcepoint_left_lateral_z = pelvis_to_leftchest_length*sind
(pelvis_to_leftchest_angle+theta_trunk_yz);
240 forcepoint_left_lateral_y = thighlocation +
pelvis_to_leftchest_length*cosd
(pelvis_to_leftchest_angle+theta_trunk_yz);
241
viscircles([forcepoint_left_lateral_z,forcepoint_left_lateral_y],.75,'EdgeCol
or',[.4 .4 .4],'LineWidth',8);
242 forcepoint_right_lateral_z = 0;
243 forcepoint_right_lateral_y = 0;
244 elseif theta_trunk_yz < 0
245 forcepoint_right_lateral_z = -pelvis_to_rightchest_length*sind
(pelvis_to_rightchest_angle-theta_trunk_yz);
246 forcepoint_right_lateral_y = thighlocation +
pelvis_to_rightchest_length*cosd
(pelvis_to_rightchest_angle-theta_trunk_yz);
247 viscircles([forcepoint_right_lateral_z,forcepoint_right_lateral_y],.
75,'EdgeColor',[.4 .4 .4],'LineWidth',8);
248 forcepoint_left_lateral_z = 0;
249 forcepoint_left_lateral_y = 0;
250 else
251 forcepoint_left_lateral_z = 0;
252 forcepoint_left_lateral_y = 0;
253 forcepoint_right_lateral_z = 0;
254 forcepoint_right_lateral_y = 0;
255 end
256
257 % Force on cushion
```

```
258 force_cushion_normal = upperbodyandthighweight-force_left_lateral*sind
(theta_trunk_yz)-force_right_lateral*sind(theta_trunk_yz)
259 force_cushion_friction = force_left_lateral*cosd(theta_trunk_yz)
+force_right_lateral*cosd(theta_trunk_yz)
260
261 % Location of force on cushion
262 forcepoint_cushion_z = (upperbodyweight*upperbodycog_yz(1)-
force_left_lateral*sind
(theta_trunk_yz)*forcepoint_left_lateral_z-force_right_lateral*sind(-
theta_trunk_yz)
*forcepoint_right_lateral_z)/force_cushion_normal;
263 forcepoint_cushion_y = thighlocation;
264 viscircles([forcepoint_cushion_z,forcepoint_cushion_y],.75,'EdgeColor',[0
.50],'LineWidth',8);
265
```